



```

XX      XX  DDDDDDD  EEEEEEEEE  LL      TTTTTTTTT  AAAAAA
XX      XX  DDDDDDD  EEEEEEEEE  LL      TTTTTTTTT  AAAAAA
XX      XX  DD        EE          LL      TT          AA      AA
XX      XX  DD        EE          LL      TT          AA      AA
  XX    XX  DD        EE          LL      TT          AA      AA
  XX    XX  DD        EE          LL      TT          AA      AA
    XX  XX  DD        EE          LL      TT          AA      AA
    XX  XX  DD        EE          LL      TT          AA      AA
  XX    XX  DD        EE          LL      TT          AA      AA
  XX    XX  DD        EE          LL      TT          AA      AA
XX      XX  DD        EE          LL      TT          AA      AA
XX      XX  DD        EE          LL      TT          AA      AA
XX      XX  DDDDDDD  EEEEEEEEE  LLLLLLLLL  TT          AA      AA
XX      XX  DDDDDDD  EEEEEEEEE  LLLLLLLLL  TT          AA      AA

```

```

LL      IIIIII  SSSSSSS
LL      IIIIII  SSSSSSS
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
LLLLLLLL  IIIIII  SSSSSSS
LLLLLLLL  IIIIII  SSSSSSS

```

(1)	51	HISTORY ; DETAILED
(1)	70	DECLARATIONS
(1)	289	PRIMARY COMMAND CHARACTER SWITCH
(1)	328	PRIMARY COMMAND SCANNER
(1)	400	ENDEXPR - END EXPRESSION
(1)	429	SLASH - OPEN CELL
(1)	460	RETURN - CLOSE CURRENT OPEN CELL
(1)	477	ENDFIELD - TERMINATE CURRENT FIELD
(1)	498	FETCH - OBTAIN DATA SPECIFIED
(1)	540	NEXTDOT - INCREMENT CURRENT LOCATION
(1)	554	OUTPUT - DISPLAY CONTENT
(1)	560	LINE FEED - DISPLAY NEXT
(1)	587	OUTPUTA - OUTPUT ADDRESS
(1)	687	GETCHAR - GET INPUT CHARACTER ROUTINE
(1)	759	PLUS/MINUS OPERATORS
(1)	779	TAB - INDIRECT DISPLAY
(1)	800	EQUALS - DISPLAY VALUE
(1)	822	SEMI - SECONDARY COMMAND SET
(1)	853	LEFT BRACKET - MODE SELECTION
(1)	877	SINGLE STEP
(1)	885	BRKPOINT - SET/CLEAR BREAKPOINTS
(1)	949	GO - START EXECUTION AT SPECIFIED LOCATION
(1)	962	SEMI-1, PC VALUE
(1)	1041	REGISTER SAVE AND RESTORE
(1)	1166	GET SCB ADDRESS
(1)	1187	BPT TRAP HANDLER
(1)	1257	TBIT EXCEPTION HANDLER
(1)	1291	UNBRK - RESTORE OPCODES FOR BREAKPOINTS
(1)	1315	SETBRK - SET BREAK POINT INSTRUCTIONS
(1)	1344	GETBPTX - GET INDEX FOR BREAKPOINT
(1)	1355	QUOTE - INPUT CHARACTER STRING
(1)	1369	DEPOSIT
(1)	1454	EXECUTE - PERFORM COMMAND STRING
(1)	1466	P - PROCESSOR REGISTER PREFIX
(1)	1474	PROCESS DEBUGGER INITIALIZATION

```

0000 1  :
0000 2  : Version:      'V04-000'
0000 3  :
0000 4  :
0000 5  :      .MCALL  MFPR
0000 6  :      .IF    DF,SW_PROCESS
0000 7  :      .TITLE  DELTA - MULTIMODE PROCESS DEBUGGER
0000 8  :      .IFF
0000 9  :      .TITLE  XDELTA - EXECUTIVE DEBUGGER
0000 10 :      .ENDC
0000 11 :      .IDENT  'V04-000'
0000 12 :
0000 13 : *****
0000 14 : *
0000 15 : *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 16 : *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 17 : *  ALL RIGHTS RESERVED.
0000 18 : *
0000 19 : *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 20 : *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 21 : *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 22 : *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 23 : *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 24 : *  TRANSFERRED.
0000 25 : *
0000 26 : *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 27 : *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 28 : *  CORPORATION.
0000 29 : *
0000 30 : *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 31 : *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 32 : *
0000 33 : *****
0000 34 :
0000 35 : ++
0000 36 : FACILITY: EXECUTIVE, DEBUGGING TOOLS
0000 37 :
0000 38 : ABSTRACT:
0000 39 : THIS MODULE PRODUCES TWO DIFFERENT DEBUGGERS DEPENDING ON THE SETTING
0000 40 : OF THE ASSEMBLY SWITCH, SW_PROCESS. DELTA IS A MULTIMODE PROCESS
0000 41 : DEBUGGER USING SYSTEM SERVICES WHILE XDELTA IS A STANDALONE EXEC
0000 42 : DEBUGGING TOOL.
0000 43 :
0000 44 : COMMAND SYNTAX IS IDENTICAL FOR BOTH VERSIONS EXCEPT FOR ENVIRONMENTAL
0000 45 : DIFFERENCES. THE SYNTAX IS QUITE TERSE AND SOMEWHAT CRYPTIC AND
0000 46 : IS DOCUMENTED IN THE "GUIDE TO WRITING AN I/O DRIVER".
0000 47 :
0000 48 : ENVIRONMENT:
0000 49 : DELTA - NORMAL PROCESS ENVIRONMENT, VARIOUS ACCESS MODES.
0000 50 : XDELTA - STANDALONE, RESIDENT, KERNEL MODE, IPL=31
0000 51 : BOTH VERSIONS MUST BE POSITION INDEPENDENT - BEWARE!
0000 52 :
0000 53 : --

```

```
0000 51 .SBTTL HISTORY ; DETAILED
0000 52 :
0000 53 : AUTHOR: R. HUSTVEDT CREATION DATE: 15-NOV-76
0000 54 :
0000 55 : REVISION HISTORY:
0000 56 :
0000 57 : V02-009 LJK0030 Lawrence J. Kenah 31-Jul-1981
0000 58 : Make changes necessary to support large physical memory
0000 59 : configurations. Change names of PFN listhead cells. Add
0000 60 : labels to cells for XE and XF stored strings to allow
0000 61 : access from INIT.
0000 62 :
0000 63 : V02-008 TCM0001 Trudy C. Matthews 29-Jul-1981
0000 64 : Change all '7ZZ's to '730's.
0000 65 :
0000 66 : V02-007 KDM0003 Kathleen D. Morse 15-Sep-1980
0000 67 : Make changes to run with multi-processor privileged program.
0000 68 :
```

```

0000 70          .SBTTL  DECLARATIONS
0000 71
0000 72  :
0000 73  : INCLUDE FILES:
0000 74  :
0000 75          $ACBDEF          : DEFINE  AST CONTROL BLOCK
0000 76          $CADEF          : DEFINE  ASSEMBLY SWITCHES
0000 77          $CLIDEF        : DEFINE  CLI VALUES
0000 78          $IPLDEF        : DEFINE  IPL VALUES
0000 79          $IRPDEF        : DEFINE  IRP VALUES
0000 80          $PCBDEF        : DEFINE  PROCESS CONTROL BLOCK
0000 81          $PRDEF         : DEFINE  PROCESSOR REGISTERS
0000 82          $PRIDEF        : DEFINE  PRIORITY INCREMENT CLASSES
0000 83          $PRTDEF        : DEFINE  PROTECTION VALUES
0000 84          $PSLDEF        : DEFINE  PSL FIELDS
0000 85          $SSDEF         : DEFINE  SYSTEM SERVICE STATUS CODES
0000 86
0000 87  :
0000 88  : MACROS:
0000 89  :
0000 90
0000 91  :
0000 92  : CPU TYPE DISPATCH MACRO:
0000 93  :
0000 94          THE ADDRESSES IN THE ADDRESS LIST ARE:
0000 95          -ADDRESS OF CODE FOR CPU TYPE=1 (11/780)
0000 96          -ADDRESS OF CODE FOR CPU TYPE=2 (11/750)
0000 97          -ADDRESS OF CODE FOR CPU TYPE=3 (11/730)
0000 98          -ADDRESS OF CODE FOR CPU TYPE=4 (?)
0000 99          -ETC.
0000 100
0000 101         CPUDISP IN INVOKED TO HANDLE CPU DIFFERENCES IN LINE.  WHEN THE
0000 102         NEXT CPU IS ADDED, ALL OCCURRENCES OF CPUDISP MUST BE EXPANDED
0000 103         TO HANDLE FOUR CPU SPECIFIC PATHS.
0000 104
0000 105         .MACRO  CPUDISP,ADDRLIST
0000 106         CASE   G^EXESGB_CPUTYPE,<ADDRLIST>,LIMIT=#PRS_SID_TYP780,TYPE=B
0000 107         .ENDM  CPUDISP
0000 108
0000 109  :
0000 110  : EQUATED SYMBOLS:
0000 111  :
00000008 0000 112 V_F1=8          : FIELD 1 PRESENT FLAG
00000009 0000 113 V_F2=9          : FIELD 2 PRESENT FLAG
0000000A 0000 114 V_F3=10         : FIELD 3 PRESENT FLAG
0000000B 0000 115 V_F4=11         : FIELD 4 PRESENT FLAG
0000000C 0000 116 V_F5=12         : FIELD 5 PRESENT FLAG
0000 117
00000000 0000 118 V_OPEN=0         : OPEN CELL FLAG
00000001 0000 119 V_ASCII=1        : ASCII
00000002 0000 120 V_INFIELD=2       : FIELD IN PROGRESS
00000003 0000 121 V_TBIT=3         : ENABLE TBIT
00000004 0000 122 V_ATBRK=4         : AT BREAKPOINT
00000005 0000 123 V_TBITOK=5       : TBIT EXPECTED
00000006 0000 124 V_RUB=6         : RUBOUT IN PROGRESS
00000007 0000 125 V_NEGATE=7       : NEGATE BIT
0000000F 0000 126 V_PMODE=15      : PROCESSOR REGISTER MCDE

```

```

0000001F 0000 127 V_PREG=71 ; PROCESSOR REGISTER FLAG
00000000 0000 128 ;
00000002 0000 129 RDCR=0 ; READ CSR
00000004 0000 130 RDBUF 2 ; READ BUFFER
00000006 0000 131 OUTCR=4 ; OUTPUT CSR
00000006 0000 132 OUTB=6 ; OUTPUT BUFFER
00000006 0000 133 ;
0000005C 0000 134 BSLSH=92 ; BACK SLASH CODE
0000000D 0000 135 CR=13 ; CARRIAGE RETURN
0000000A 0000 136 LF=10 ; LINE FEED
00000027 0000 137 QUOT=39 ; QUOTE
0000007F 0000 138 RUBOUT=127 ; RUBOUT CODE
0000002F 0000 139 SLSH=47 ; SLASH CODE
00000000 0000 140 ;
00000000 0000 141 ;
00000000 0000 142 ;
00000000 0000 143 :
00000000 0000 144 : OWN STORAGE:
00000000 0000 145 :
00000000 0000 146 .LIST MEB ; DISPLAY MACRO EXPANSIONS
00000000 0000 147 .IF DF,SW_PROCESS
00000000 0000 148 .PSECT DELTA, LONG
00000000 0000 149 DELBASE: .LONG DELBASE-DELBASE ; RELATIVE PAGE NUMBER OF WRITABLE
00000000 0000 150 .LONG <511+DELEND-DELBASE>&^C511 ; REL PAGE NUMBER OF END OF WRITABLE
00000000 0000 151 .LONG DELTA_START-DELBASE ; START ADDRESS
00000000 0000 152 .IFF
00000000 0000 153 .PSECT Z$DEBUGXDELTA, LONG
00000000 0000 154 .ENDC
00000000 0000 155 CONTEXT: ;
00000000 0000 156 .LONG 0 ; BUFFER PADDING
00000034 0004 157 INBUF: .BLKB 48 ; INPUT BUFFER
00000000 0034 158 STATUS: .LONG 0 ; STATUS FLAGS
00000000 0038 159 F1: .LONG 0 ; FIELDS
00000000 003C 160 F2: .LONG 0 ; 0-7
00000000 0040 161 F3: .LONG 0 ;
00000000 0044 162 F4: .LONG 0 ;
00000000 0048 163 F5: .LONG 0 ;
00000000 004C 164 ;
00000000 004C 165 MFYFLG: .LONG 0 ; MODIFY ENABLE FLAG FOR OTHER PROCESS
00000000 0050 166 ; ADDRESS SPACES
00000000 0050 167 PID: .LONG 0 ; PID FOR ADDRESS SPACE 0=>SELF
00000000 0054 168 ;
00000000 0054 169 FCTR: .BYTE 0 ; FIELD COUNTER
00000000 0055 170 ;
00000000 0055 171 DTYPE: .BYTE 2 ; DATA TYPE
00000000 0056 172 CURTYPE: .BYTE 2 ; CURRENT TYPE
00000000 0057 173 ;
00000000 0057 174 OPER: .BYTE 0 ; OPERATOR
00000000 0058 175 B: ; BASE OF DATA AREA(CENTER)
00000000 0058 176 CURDOT: .LONG 0 ; CURRENT LOCATION
00000000 005C 177 QUAN: .LONG 0 ; QUANTITY (:Q)
00000070 0060 178 OUTBUF: .BLKL 4 ; OUTPUT BUFFER
00000070 0070 179 ;
00000070 0070 180 : REGISTER SAVE AREA
00000070 0070 181 :
00000074 0070 182 SAVREG: ; REGISTER SAVE AREA
00000074 0070 183 .BLKL 1 ; R0

```

```

00000078 0074 184          .BLKL 1          : R1
0000007C 0078 185 SAVR2: .BLKL 1          : R2
00000080 007C 186          .BLKL 1          : R3
00000084 0080 187          .BLKL 1          : R4
00000088 0084 188          .BLKL 1          : R5
0000008C 0088 189          .BLKL 1          : R6
00000090 008C 190          .BLKL 1          : R7
00000094 0090 191          .BLKL 1          : R8
00000098 0094 192          .BLKL 1          : R9
0000009C 0098 193          .BLKL 1          : R10
000000A0 009C 194          .BLKL 1          : R11
000000A4 00A0 195 SAVAP: .BLKL 1          : AP
000000A8 00A4 196          .BLKL 1          : (FP)
000000AC 00A8 197 SAVSP: .BLKL 1          : SP
000000B0 00AC 198 SAVPC: .BLKL 1          : PC
000000B4 00B0 199 SAVPSL: .BLKL 1         : PSL
000000B6 00B4 200 SAVOCR: .BLKW 1         : OUTPUT CSR SAVE
000000B8 00B6 201 SAVRCR: .BLKW 1         : INPUT CSR SAVE
000000B8 00B8 202 ASTEN:          : AST ENABLE SAVE LOCATION
000000BC 00B8 203 SAVRXCS: .BLKL 1         : CONSOLE RECEIVER STATUS
000000BC 00BC 204          :
000000BC 00BC 205 CONTEXTSZ=-CONTEXT          : SIZE OF PER MODE CONTEXT AREA
000000BC 00BC 206          :
000000BC 00BC 207          : RESERVE SPACE FOR MULTIPLE MODE CONTEXT AREA
000000BC 00BC 208          :
000000BC 00BC 209          .IF DF,SW_PROCESS          :
000000BC 00BC 210          .REPT 3          :
000000BC 00BC 211          .BLKB CONTEXTSZ          : FOR EXEC,SUPER AND USER
000000BC 00BC 212 SAV...= .          :
000000BC 00BC 213          .=-CONTEXTSZ+<DTYPE-CONTEXT>          : POINT AT DTYPE,CURTYP
000000BC 00BC 214          .BYTE 2,2          : SET TYPE TO LONGWORD
000000BC 00BC 215          .=SAV...          : RESTORE LOCATION COUNTER
000000BC 00BC 216          .ENDR          :
000000BC 00BC 217          .ENDC          :
000000BC 00BC 218          :
000000BC 00BC 219          :
000000BC 00BC 220          :
000000BC 00BC 221          : BREAK POINT DATA
000000BC 00BC 222          :
000000B8 00BC 223 BRKADR=-4          :
000000BC 00BC 224          .IF NDF,SW_PROCESS          :
00000000' 00BC 225 XDELIBRK::          :
00000000' 00BC 226          .LONG INISBRK          : ADDRESS OF INITIAL BREAKPOINT
00000000' 00C0 227          .IFF          : FOR PROCESS VERSION
00000000' 00C0 228 INIBRKA: .LONG 0          : INITIAL BREAKPOINT
00000000' 00C0 229          .ENDC          :
000000DC 00C0 230          .BLKL 7          : OTHER BREAK POINT ADDRESSES
00000008 00DC 231 NBRK=<.-4-BRKADR>/4          : NUMBER OF BREAKPOINTS
000000DB 00DC 232 BRKOP=-1          : SAVED OPCODE
000000E4 00DD 233          NOP          : INITIAL OPCODE
000000E4 00E4 234          .BLKB 7          : REMAINING OPCODES
000000E4 00E4 235          :
000000E4 00E4 236          :
000000E0 00E4 237 BRKDSP=-4          :
00000104 00E4 238          .BLKL 8          : DISPLAY LOCATION START
00000100 0104 239 BRKCOM=-4          :
00000124 0104 240          .BLKL 8          : COMMAND START

```



```

00000130 0124 241
0124 242 XREGV: .BLKL 3 ; X REGISTER VECTOR
0130 243 XDEL_LOADBASE:: ; BASE OF LOADABLE
00000000 0130 244 .LONG 0 ; CPU DEPENDENT CODE
00000000 0134 245 .LONG SCH$GL_CURPCB ; X4 = CURRENT PCB ADDRESS
00000000 0138 246 .LONG SCH$GL_PCBVEC ; X5 = BASE OF PCB VECTOR
013C 247 .IF NDF_SW_PROCESS
00000000 013C 248 .LONG PFNSAW_SWPVBN ; X6 = SWAP VBN
00000000 0140 249 .LONG PFNSAL_PTE ; X7 = PTE BACK POINTER
00000000 0144 250 .LONG PFNSAL_BAK ; X8 = BACKUP ADDRESS
00000000 0148 251 .LONG PFNSAW_REFCNT ; X9 = REFERENCE COUNT
00000000 014C 252 .LONG PFNSAx_FLINK ; XA = FORWARD LINK
00000000 0150 253 .LONG PFNSAx_BLINK ; XB = BACK LINK
00000000 0154 254 .LONG PFNSAB_STATE ; XC = STATE
00000000 0158 255 .LONG PFNSAB_TYPE ; XD = TYPE
015C 256 XD$SGL_XESTRING::
0000000 015C 257 .LONG XD$SGT_WORD_PFN ; XE;E WITH X0 = PFN , DEFAULT TO WORD ARRAY
0160 258 XD$SGL_XFSTRING::
00000000 0160 259 .LONG XD$SGT_WORD_PFN ; XF;E WITH R0 = PFN , DEFAULT TO WORD ARRAY
00000168 0164 260 MCHKSAV: .BLKL 1 ; SAVED CONTENT OF MACHINE CHECK VECTOR
0168 261 .IFF ; FOR PROCESS VERSION
0168 262 .BLKL 10
0168 263 TTIOSB: .BLKL 2 ; IO STATUS BLOCK FOR TERMINAL READ
0168 264 TTCHAN: .BLKL 1 ; CHANNEL NUMBER
0168 265 TTNAMD: .LONG 2,TTSTR ; ACTUAL ADDRESS FOR DESCR SET BY INIT
0168 266 TTSTR: .ASCII /TT/
0168 267 DBGACTIVE: ; ACTIVE FLAGS BY ACCESS MODE
0168 268 .LONG 0
0168 269 EXITBLK: ; EXIT HANDLER BLOCK
0168 270 .LONG 0
0168 271 EXIHADR: .LONG EXIHANDLE ; EXIT HANDLER
0168 272 .LONG 1 ; ARGUMENT COUNT
0168 273 EXCODA: .LONG EXITCODE ; ADDRESS TO STORE EXIT CODE
0168 274 EXITCODE:
0168 275 .LONG 1 ; RECEIVER FOR EXIT CODE
0168 276 KCOND: .LONG 0 ; PREVIOUS KERNEL HANDLER
0168 277 ECOND: .LONG 0 ; PREVIOUS EXEC HANDLER
0168 278 SCOND: .LONG 0 ; PREVIOUS SUPER HANDLER
0168 279 TERMASKD: ; TERMINATOR MASK DESCRIPTOR
0168 280 .LONG 16 ; MASK LENGTH
0168 281 .LONG TERMASK ; MASK ADDRESS
0168 282 TERMASK: .LONG <1a9>!<1a10>!<1a13>!<1a27> ; TAB,LF,CR,ESC
0168 283 .LONG <1a2>!<1a15>!<1a29> ; DOUBLE QUOTE,SLASH,EQUALS
0168 284 .LONG <1a19> ; 'S'
0168 285 .LONG 0
0168 286
0168 287 .ENDC

```

```

                                .SBTTL PRIMARY COMMAND CHARACTER SWITCH
                                0168 289
                                0168 290
                                0168 291 :
                                0168 292 : PRIMARY CHARACTER LIST
                                0168 293 :
                                0168 294 PRIMARY:
42 41 39 38 37 36 35 34 33 32 31 30 0168 295 .ASCII /0123456789ABCDEF/ ; DECIMAL AND HEX CHARS
                                46 45 44 43 0174
                                2E 0178 296 .ASCII /./ ; DOT - CURRENT LOCATION
                                2C 0179 297 .ASCII /,/ ; COMMA - FIELD SEPARATOR
                                00000012 017A 298 OPERBAS=-PRIMARY ; OPERATORS
                                2B 017A 299 .ASCII /+ / ; PLUS - ADD
                                20 017B 300 .ASCII / / ; BLANK - SAME AS PLUS
                                40 017C 301 .ASCII /@ / ; SHIFT OPERATOR
                                2A 017D 302 .ASCII /* / ; MULTIPLY OPERATOR
                                25 017E 303 .ASCII /% / ; DIVIDE OPERATOR
                                2D 017F 304 .ASCII /- / ; MINUS - SUBTRACT OPERATOR
                                5B 0180 305 .ASCII /[ / ; LBRACKET - LEFT BRACKET
                                0181 306 TERM: ; BASE OF TERMINATOR LIST
                                09 0181 307 .ASCII <9> ; TAB - INDIRECT
                                0A 0182 308 .ASCII <10> ; LINEFEED -
                                0D 0183 309 .ASCII <CR> ; RETURN -
                                2F 0184 310 .ASCII '/' ; SLASH - OPEN FOR DISPLAY
                                22 0185 311 .ASCII '...' ; DOUBLE QUOTE - OPEN FOR ASCII DISPLAY
                                3D 0186 312 .ASCII /= / ; EQUALS - DISPLAY
                                1B 0187 313 .ASCII <27> ; ESCAPE - PREVIOUS LOCATION
                                53 0188 314 .ASCII /S / ; STEP
                                00000008 0189 315 NTERM=-TERM ; NUMBER OF TERMINATORS
                                38 0189 316 .ASCII <59> ; SEMI - INITIATE SECONDARY
                                3A 018A 317 .ASCII /:/ ; COLON - SEPARATE PID FORM ADDRESS
                                50 018B 318 .ASCII /P / ; P - PROCESSOR REGISTER PREFIX
                                51 018C 319 .ASCII /Q / ; Q - LAST QUANTITY
                                27 018D 320 .ASCII /' / ; QUOTE - BEGIN CHAR STRING
                                52 018E 321 .ASCII /R / ; REGISTER PREFIX
                                47 018F 322 .ASCII /G / ; G - GLOBAL PREFIX
                                48 0190 323 .ASCII /H / ; H - HIGH, P1 SPACE PREFIX
                                58 0191 324 .ASCII /X / ; X REGISTER PREFIX
                                0000002A 0192 325 NPRIM=-PRIMARY ; NUMBER OF PRIMARY COMMANDS
                                0192 326

```

```

0192 328 .SBTTL PRIMARY COMMAND SCANNER
0192 329
0192 330 :
0192 331 :
0192 332 :
0192 333 :
0192 334
0192 335 OUTER: .ASCIZ <LF><CR>/EH?/<LF><CR>
019A 336
019A 337 DCOM: .WORD : CALL ENTRY POINT
019C 338 .IF DF,SW PROCESS : FOR PROCESS VERSION ONLY
019C 339 MOVAB W^DBGEXCEP,(FP) : SET CONDITION HANDLER ADDRESS
019C 340 .ENDC
019C 341 BRB SCANP : ENTER SCANP
54 F1 AF 9E 019E 342 ERROR: MOVAB OUTER,R4 : SET ADDR OF CONTROL STRING
01A2 343 BSBW OUTZSTRING : OUTPUT ASCIZ STRING
59 SE 5D D0 01A5 344 SUPERST:MOVL FP,SP : RESET STACK
AC AB 9E 01AB 345 MOVAB INBUF-B(R11),R9 : RESET STRING ADDRESS
69 94 01AC 346 CLRB (R9) : AND FORCE READ
02C6 30 01AE 347 BSBW RESET : RESET SCANNER
02 10 01B1 348 SCANP: BSBW NEXTP : SCAN INPUT
FC 11 01B3 349 BRB SCANP : SCAN IT ALL
01B5 350 NEXTP: : PROCESS NEXT PRIMARY CHAR
01B5 351 BSBW GETCHAR : GET CHARACTER
AB AF 2A 58 3A 01B8 352 LOCC RB,#NPRIM,PRIMARY : CHECK IT
DF 13 01BD 353 BEQL ERROR : NOT FOUND, ERROR
50 2A 50 C3 01BF 354 SUBL3 RO,#NPRIM,RO : RATIONALIZE INDEX
01C3 355 CASE RO,LIMIT=#16,<-
01C3 356 DOT,- : DOT - CURRENT LOCATION
01C3 357 COMMA,- : COMMA - FIELD SEPARATOR
01C3 358 OPERATOR,- : OPERATOR -
01C3 359 OPERATOR,- : PLUS - ADD OPERATOR
01C3 360 OPERATOR,- : BLANK - ADD OPERATOR
01C3 361 OPERATOR,- : @ - SHIFT OPERATOR
01C3 362 OPERATOR,- : * - MULTIPLY OPERATOR
01C3 363 OPERATOR,- : % - DIVIDE OPERATOR
01C3 364 NEGATE,- : MINUS - SUBTRACT/NEGATE
01C3 365 LBRACKET,- : LEFT BRACKET - MODE SELECT
01C3 366 TAB,- : TAB - INDIRECT
01C3 367 LINEFEED,- : LINE FEED - NEXT LOCATION
01C3 368 RETURN,- : RETURN - CLOSE OPEN CELL
01C3 369 SLASH,- : SLASH - OPEN FOR DISPLAY
01C3 370 DQUOTE,- : DOUBLE QUOTE - OPEN FOR ASCII DISPLAY
01C3 371 EQUALS,- : EQUALS - DISPLAY VALUE
01C3 372 ESCAP,- : ESCAPE - PREVIOUS LOCATION
01C3 373 STEP,- : 'S' - SINGLE STEP
01C3 374 SEMI,- : SEMI COLON - SECONDARY COMMAND
01C3 375 COLON,- : COLON - SEPARATE PID FROM ADDRESS
01C3 376 PREG,- : 'P' - PROCESSOR REGISTER
01C3 377 QUANT,- : 'Q' - LAST QUANTITY
01C3 378 REGISTER,- : QUOTE - BEGIN ASCII STRING
01C3 379 GLOBL,- : G - GLOBAL PREFIX
01C3 380 HIGH,- : H - P1 SPACE PREFIX
01C3 381 XREG,- : X REGISTER
19' 10 50 AF 01C3 382 >
30000$: CASEW RO,#16,S^#<<30001$-30000$>/2>-1
01C7

```

```

03C8' 01C7      .SIGNED_WORD  DOT-30000$
00D4' 01C9      .SIGNED_WORD  COMMA-30000$
026A' 01CB      .SIGNED_WORD  OPERATOR-30000$
026A' 01CD      .SIGNED_WORD  OPERATOR-30000$
026A' 01CF      .SIGNED_WORD  OPERATOR-30000$
026A' 01D1      .SIGNED_WORD  OPERATOR-30000$
026A' 01D3      .SIGNED_WORD  OPERATOR-30000$
0273' 01D5      .SIGNED_WORD  NEGATE-30000$
02EB' 01D7      .SIGNED_WORD  LBRACKET-30000$
027F' 01D9      .SIGNED_WORD  TAB-30000$
012B' 01DB      .SIGNED_WORD  LINEFEED-30000$
00BD' 01DD      .SIGNED_WORD  RETURN-30000$
008C' 01DF      .SIGNED_WORD  SLASH-30000$
0087' 01E1      .SIGNED_WORD  DQUOTE-30000$
02A1' 01E3      .SIGNED_WORD  EQUALS-30000$
028F' 01E5      .SIGNED_WORD  ESCAP-30000$
0306' 01E7      .SIGNED_WORD  STEP-30000$
02C4' 01E9      .SIGNED_WORD  SEMI-30000$
03B8' 01EB      .SIGNED_WORD  COLON-30000$
065E' 01ED      .SIGNED_WORD  PREG-30000$
03D5' 01EF      .SIGNED_WORD  QUANT-30000$
0616' 01F1      .SIGNED_WORD  QUOTE-30000$
03ED' 01F3      .SIGNED_WORD  REGISTER-30000$
0043' 01F5      .SIGNED_WORD  GLOBL-30000$
0049' 01F7      .SIGNED_WORD  HIGH-30000$
041B' 01F9      .SIGNED_WORD  XREG-30000$
          01FB      30001$:
          10  50  B1  01FB  383  CMPW   R0,#16      ; IS NUMBER > RADIX
          9E  18  01FE  384  BGEQ   ERROR      ; YES
          56  10  C4  0200  385  MULL   #16,R6      ; SCALE BY RADIX
          56  50  C0  0203  386  ADDL   R0,R6      ; AND ADD NEW DIGIT
          6A  04  C8  0206  387  INFLD: BISL   #<1@V_INFIELD>,(R10) ; NOTE FIELD INPUT
          05  0209  388  RSB                    ; NEXT PRIMARY CHARACTER
          020A  389
          020A  390
          54  01  1F  9C  020A  391  GLOBL: ROTL   #31,#1,R4    ; GENERATE SYSTEM SPACE PREFIX
          07  11  020E  392  BRB     PRE1      ; MERGE WITH COMMON
54  7FFE0000  8F  D0  0210  393  HIGH:  MOVL  #^X7FFE0000,R4 ; P1 SPACE BASE ADDRESS
          06  10  0217  394  PRE1:  BSBB  ENDEXPR ; END EXPRESSION
          56  54  D0  0219  395  MOVL  R4,R6      ; SET INTO ACCUM
          E7  AF  9F  021C  396  PUSHAB INFLD ; RETURN THROUGH INFLD
          021F  397  ;
          021F  398  BRB   ENDEXPR

```

```

021F 400 .SBTTL ENDEXPR - END EXPRESSION
021F 401
021F 402 :
021F 403 :
021F 404 :
021F 405 ENDEXPR:
03 6A 07 E5 021F 406 BBCC #V_NEGATE,(R10),5$ ; SKIP IF NOT NEGATE
56 56 CE 0223 407 MNEGL R6-R6 ; NEGATE ACCUMULATOR
06 10 0226 408 5$: BSBB 10$ ; PERFORM OPERATION
56 D4 0228 409 CLRL R6 ; CLEAR ACCUMULATOR
FF AB 94 022A 410 CLRB OPER-B(R11) ; INIT OPERATOR
05 022D 411 RSB ; AND RETURN
022E 412 10$: CASE OPER-B(R11),TYPE=B,<- ; DO OPERATION
022E 413 ADD,- ; ADD, PLUS
022E 414 ADD,- ; BLANK, PLUS
022E 415 SHFT,- ; SHIFT, @
022E 416 MUL,- ; MULTIPLY, *
022E 417 DIV,- ; DIVIDE, %
022E 418 >
04' 00 FF AB 8F 022E 419 CASEB OPER-B(R11),#0,S^#<<30003$-30002$>/2>-1
0233 30002$:
0017' 0233 .SIGNED_WORD ADD-30002$
0017' 0235 .SIGNED_WORD ADD-30002$
000A' 0237 .SIGNED_WORD SHFT-30002$
000F' 0239 .SIGNED_WORD MUL-30002$
0013' 023B .SIGNED_WORD DIV-30002$
023D 30003$:
57 57 56 78 023D 419 SHFT: ASHL R6,R7,R7 ; SHIFT
05 0241 420 RSB ; AND EXIT
57 56 C4 0242 421 MUL: MULL R6,R7 ; MULTIPLY
05 0245 422 RSB ; AND EXIT
57 56 C6 0246 423 DIV: DIVL R6,R7 ; DIVIDE
05 0249 424 RSB ; AND EXIT
57 56 C0 024A 425 ADD: ADDL R6,R7 ; ADD
05 024D 426 RSB ; AND EXIT
024E 427

```

```

024E 429 .SBTTL SLASH - OPEN CELL
024E 430
024E 431 :
024E 432 :
024E 433 :
024E 434 DQUOTE: : DISPLAY IN ASCII
024E 435 BISB #<1@V_ASCII>,(R10) : SET ASCII FLAG
0251 436 BRB OPEN :
0253 437
0253 438 SLASH: :
0253 439 BICB #<1@V_ASCII>,(R10) : CLEAR ASCII; DISPLAY MODE
0256 440 OPEN: BSBB ENDFIELD : TERMINATE FIELD
06 6A 08 E0 0258 441 BBS #V F1,(R10),5$ : ADDR SPECIFIED?
6B 04 AB D0 025C 442 MOVL QUAN-B(R11),CURDOT-B(R11) : ; NO, GO INDIRECT
04 11 0260 443 BRB 10$ : AND DISPLAY CONTENT
6B E0 AB D0 0262 444 5$: MOVL F1-B(R11),CURDOT-B(R11) : SET NEW DOT
50 6A 01 0F EF 0266 445 10$: EXTZV #V PRMODE,#1,(R10),R0 : GET PROCESSOR REGISTER MODE FLAG
6A 01 1F 50 F0 026B 446 RO,#V PREG,#1,(R10) : AND MOVE TO SEMI-PERMANENT COPY
0086 30 0270 447 BSBW LOCOUT : OUTPUT AND OPEN
1A 6A 09 E1 0273 448 BBC #V F2,(R10),RSET : RANGE SPECIFIED?
6B E4 AB D1 0277 449 15$: CMPL F2-B(R11),CURDOT-B(R11) : CHECK FOR END
14 15 027B 450 BLEQ RSET : YES
027D 451 .IF NDF,SW_PROCESS :
76 10 027D 452 BSBB NEXTLOC : INCREMENT TO NEXT DOT
027F 453 .IFF :
027F 454 BSBW NEXTLOC : INCREMENT TO NEXT DOT
027F 455 .ENDC :
F6 11 027F 456 BRB 15$ : AND CONTINUE
FF1A 31 0281 457 ERR4: BRW ERROR : DECLARE ERROR
0284 458

```

```

0284 460 .SBTTL RETURN - CLOSE CURRENT OPEN CELL
0284 461
0284 462 :
0284 463 : RETURN - CLOSE CURRENT OPEN CELL
0284 464 :
0284 465
0284 466 RETURN:
0284 467 BSBB ENDFIELD : TERMINATE CURRENT FIELD
0286 468 .ENABL LSB :
0A 6A 00 E5 0286 469 BBCC #V_OPEN,(R10),10$ : SKIP IF NONE OPEN
03 6A 08 E1 028A 470 BBC #V_F1,(R10),RSET : SKIP IF NOTHING TO STORE
0560 30 028E 471 BSBW DEPOSIT : DEPOSIT
01E3 31 0291 472 RSET: BRW RESET : RESET SCANNER
F9 6A 08 E1 0294 473 10$: BBC #V_F1,(R10),RSET : DONE IF NO INPUT
01D4 31 0298 474 BRW EQ[1 : OTHERWISE OUTPUT
029B 475 .DSABL LSB :

```

```

029B 477 .SBTTL ENDFIELD - TERMINATE CURRENT FIELD
029B 478
029B 479 :
029B 480 : COMMA TERMINATE CURRENT FIELD
029B 481 :
FF68 30 029B 482 COMMA: BSBW INFLD ; ZERO IF NULL FIELD
029E 483
029E 484 :
029E 485 : TERMINATE CURRENT FIELD
029E 486 :
029E 487 ENDFIELD:
16 6A 02 E5 029E 488 BBCC #V INFIELD,(R10),10$ ; CLEAR PENDING FIELD
FF7A 30 02A2 489 BSBW ENDEXPR ; END EXPRESSION
50 FC AB 9A 02A5 490 MOVZBL FCTR-B(R11),R0 ; GET FIELD POINTER
D3 01 AA 50 E2 02A9 491 BBSS R0,1(R10),ERR4 ; ERROR IF TOO MANY FIELDS
EO AB40 57 D0 02AE 492 MOVL R7,F1-B(R11)[R0] ; STORE FIELD VALUE
FC AB 96 02B3 493 INCB FCTR-B(R11) ; INCREMENT FIELD COUNTER
56 7C 02B6 494 CLRQ R6 ; CLEAR ACCUMULATORS
05 02B8 495 10$: RSB ; RETURN
02B9 496

```



```

02B9 498 .SBTTL FETCH - OBTAIN DATA SPECIFIED
02B9 499
02B9 500 :
02B9 501 :
02B9 502 :
02B9 503 :
02B9 504 :
02BD 505 :
02BD 506 :
02BD 507 :
02BD 508 :
02BD 509 :
02BD 510 :
02BD 511 :
02BD 512 :
02BD 513 :
1D 6A 1F E0 02B9 504 : BBS #V_PREG,(R10),40$ : BR IF PROCESSOR REGISTER
02BD 505 : .IF DF,SW_PROCESS :
02BD 506 : TSTL PID-B(R11) : CHECK FOR PROCESS GET
02BD 507 : BNEQ 50$ : BR IF YES
02BD 508 : .ENDC
02BD 509 : CASE CURTYPE-B(R11),TYPE=B,<- : OPERATE ON TYPE
02BD 510 : 10$,- : BYTE
02BD 511 : 20$,- : WORD
02BD 512 : 30$,- : LONG
02BD 513 : >
02' 00 FE AB 8F 02BD 514 : CASEB CURTYPE-B(R11),#0,S^#<<30005$-30004$>/2>-1
02C2 30004$:
0006' 02C2 : .SIGNED_WORD 10$-30004$
000C' 02C4 : .SIGNED_WORD 20$-30004$
0012' 02C6 : .SIGNED_WORD 30$-30004$
02C8 30005$:
04 AB 00 BB 9A 02C8 514 10$: MOVZBL @CURDOT-B(R11),QUAN-B(R11) : GET BYTE
05 02CD 515 : RSB : RETURN
04 AB 00 BB 3C 02CE 516 20$: MOVZWL @CURDOT-B(R11),QUAN-B(R11) : GET WORD
05 02D3 517 : RSB : RETURN
04 AB 00 BB D0 02D4 518 30$: MOVL @CURDOT-B(R11),QUAN-B(R11) : GET LONGWORD
05 02D9 519 : RSB : RETURN
02DA 520 : .IF NDF,SW_PROCESS :
02DA 521 40$: MFPR CURDOT-B(R11),QUAN-B(R11) : GET PROCESSOR REGISTER
04 AB 5B DB 02DA : MFPR CURDOT-B(R11),QUAN-B(R11)
05 02DE 522 : RSB
02DF 523 : .IFF : FALSE IF PROCESS VERSION
02DF 524 40$: $CMKRNLS B^FTCHPREG,(AF) : CALL IN KERNEL MODE TO FETCH
02DF 525 : RSB
02DF 526 : BRW FTCHP : FETCH FROM FOREIGN PROCESS
02DF 527 50$: .ENDC
02DF 528 :
02DF 529 :
02DF 530 : .IF DF,SW_PROCESS :
02DF 531 FTCHPREG: :
02DF 532 : .WORD 0 : ENTRY MASK
02DF 533 : MOVAB W^PREXC,(FP) : SET EXCEPTION HANDLER
02DF 534 : MFPR CURDOT-B(R11),QUAN-B(R11) : GET PROCESSOR REGISTER
02DF 535 : MOVL #1,R0 : RETURN SUCCESS
02DF 536 : RET :
02DF 537 :
02DF 538 : .ENDC :

```

```

02DF 540 .SBTTL NEXTDOT - INCREMENT CURRENT LOCATION
02DF 541
02DF 542
02DF 543 :
02DF 544 : INCREMENT TO NEXT LOCATION
02DF 545 NEXTDOT:
02DF 546 MOVL #1,R1 : ASSUME UNIT INCREMENT
51 01 D0 02DF 547 TSTL (R10) : CHECK FOR PREG
6A D5 02E2 548 BLSS 10$ : YES, USE UNIT INCREMENT
05 19 02E4 549 ROTL CURTYPE-B(R11),R1,R1 : FORM INCREMENT
51 51 FE AB 9C 02E6 550 10$: ADDL R1,CURDOT-B(R11) : AND ADD TO DOT
6B 51 C0 02EB 551 RSB : RETURN
05 02EE 552
02EF 552

```

```

02EF 554 .SBTTL OUTPUT - DISPLAY CONTENT
02EF 555 :
02EF 556 : OUTPUT CONTENT
02EF 557 :
02EF 558 OUTBB:
1C 0C 04 02EF 559 .BYTE 4,12,28 ; STARTING DIGIT LIST
02F2 560 .SBTTL LINE FEED - DISPLAY NEXT
02F2 561 :
02F2 562 :
02F2 563 :
02F2 564 LINEFEED:
FF8F 30 02F2 565 BSBW RETURN ; CLOSE OPEN CELL
02F5 566 NEXTLOC: ; PROMPT WITH NEXT LOCATION
E8 10 02F5 567 BSBB NEXTDOT ; INCREMENT LOCATION
02F7 568 LOC PROMPT: ; DISPLAY ADDR/CONTENT
2B 10 02F7 569 BSBB OUTPUTA ; OUTPUT ADDRESS
BE 10 02F9 570 LOCOUT: BSBB FETCH ; FETCH CONTENT
6A 01 C8 02FB 571 BISL #<1@V_OPEN>, (R10) ; INDICATE OPEN CELL
02FE 572
02FE 573 OUTPUT:
51 FE AB 9A 02FE 574 MOVZBL CURTYPE-B(R11),R1 ; GET TYPE
52 E9 AF41 9A 0302 575 MOVZBL OUTBB[R1],R2 ; INIT DIGIT SELECTOR
53 04 AB D0 0307 576 MOVL QUAN-B(R11),R3 ; GET QUANTITY TO DISPLAY
04 6A 01 E0 030B 577 BBS #V ASCII,(R10),10$ ; CHECK FOR ASCII OUT
53 10 030F 578 BSBB OUTCOM ; OUTPUT NUMBER IN HEX
0E 11 0311 579 BRB 20$ ; AND EXIT THROUGH OUTSPACE
08 AB 53 D0 0313 580 10$: MOVL R3,OUTBUF-B(R11) ; PUT STRING IN BUFFER
52 01 51 78 0317 581 ASHL R1,#1,R2 ; GET COUNT
08 AB42 94 031B 582 CLRB OUTBUF-B(R11)[R2] ; MARK END OF STRING
59 10 031F 583 BSBB OUTZBUF ; OUTBUF ASCIIZ BUFFER
008B 31 0321 584 20$: BRW OUTSPACE ; FOLLOW WITH SPACE
0324 585

```

```

0324 587 .SBTTL OUTPUTA - OUTPUT ADDRESS
0324 588 :
0324 589 : OUTPUT ADDRESS
0324 590 :
0324 591 OUTPUTA:
0324 592 BSBW CRLF : OUTPUT ADDRESS
53 008D 30 0327 593 MOVAB SAVREG-B(R11),R3 : OUTPUT CR/LF
18 AB 9E 0328 594 .IF DF,SW_PROCESS : BASE OF REGISTER AREA
0328 595 TSTL PID-B(R11) : ONLY FOR PROCESS VERSION
0328 596 BNEQ 3$ : CHECK FOR OTHER PROCESS ADDRESS
0328 597 .ENDC : BR IF YES
53 6B 53 C3 0328 598 SUBL3 R3,CURDOT-B(R11),R3 : COMPUTE OFFSET INTO REGISTER AREA
12 19 032F 599 BLSS 5$ : NOT GENERAL REGISTER
53 04 C6 0331 600 DIVL #4,R3 : SCALE TO LONGWORD NUMBER
OF 53 D1 0334 601 CMPL R3,#15 : CHECK FOR MAX REG NUMBER
0A 14 0337 602 BGTR 5$ : GTR, NOT A REGISTER
50 52 8F 9A 0339 603 MOVZBL #^A'R',RO : OUTPUT PREFIX
52 10 033D 604 BSBB OUTCHAR : OF 'R'
52 D4 033F 605 CLRL R2 : AND SET FOR ONE DIGIT OF OUTPUT
13 11 0341 606 BRB 10$
0343 607 .IF DF,SW_PROCESS : FOR PROCESS VERSION ONLY
0343 608 3$: TSTL (R10)- : CHECK FOR PROCESSOR REGISTER
0343 609 BLSS 5$ : BR IF YES
0343 610 MOVL #28,R2 : SET FOR LONGWORD OUTPUT
0343 611 MOVL PID-B(R11),R3 : GET PID OF TARGET
0343 612 BSBB OUTCOM : OUTPUT PID AS LONGWORD
0343 613 MOVZBL #^A':',RO : SEPARATE WITH ':'
0343 614 BSBB OUTCHAR : OUTPUT COLON
0343 615 .ENDC
53 6B D0 0343 616 5$: MOVL CURDOT-B(R11),R3 : GET ADDRESS
52 1C D0 0346 617 MOVL #28,R2 : ASSUME LONGWORD OUTPUT
6A D5 0349 618 TSTL (R10) : CHECK FOR PROCESSOR REGISTER
50 50 09 18 034B 619 BGEQ 10$ : NO, JUST A LONGWORD
8F 9A 034D 620 MOVZBL #^A'P',RO : PRECEDE WITH A 'P'
3E 10 0351 621 BSBB OUTCHAR : OUTPUT P
52 04 D0 0353 622 MOVL #4,R2 : SET FIELD TO 2 DIGITS
0C 10 0356 623 10$: BSBB OUTCOM : COMMON OUTPUT
50 2F 9A 0358 624 MOVZBL #SLSH,RO : OUTPUT SLASH
34 11 035B 625 BRB OUTCHAR : RETURN THROUGH OUTCHAR
035D 626 OUTDIGIT: : OUTPUT ONE DIGIT
52 D4 035D 627 CLRL R2 : ZAP DIGIT SELECTOR
03 11 035F 628 BRB OUTCOM : AND MERGE WITH COMMON
0361 629
0361 630 OUTLONG: : OUTPUT LONGWORD
52 1C D0 0361 631 MOVL #28,R2 : SET DIGIT SELECTOR
0364 632 OUTCOM: : FORMAT IT
54 08 AB 9E 0364 633 MOVAB OUTBUF-B(R11),R4 : GET ADDRESS OF OUTPUT BUFFER
51 53 04 52 EF 0368 634 10$: EXTZV R2,#4,R3,R1 : GET DIGIT
84 FDF6 CF41 90 036D 635 MOVB PRIMARY[R1],(R4)+ : BUFFER IT
52 04 C2 0373 636 SUBL #4,R2 : NEXT DIGIT
F0 18 0376 637 BGEQ 10$ : DO ALL REQUESTED
64 94 0378 638 CLRB (R4) : MARK END OF BUFFER
54 08 AB 9E 037A 639 OUTZBUF:MOVAB OUTBUF-B(R11),R4 : GET START OF BUFFER
037E 640
037E 641 OUTZSTRING: : OUTPUT ASCIZ STRING
50 84 9A 037E 642 MOVZBL (R4)+,RO : GET A CHAR
04 13 0381 643 BEQL 10$ : BR IF DONE

```

	0C	10	0383	644	BSBB	OUTCHAR	: OUTPUT CHAR
	F7	11	0385	645	BRB	OUTZSTRING	: CONTINUE
		05	0387	646	10\$: RSB		: RETURN IF DONE
			0388	647			
			0388	648			
50	5C	8F	9A	0388	649	OUTBSLSH:	: OUTPUT BACK SLASH
		03	11	0388	650	MOVZBL	: SET CHARACTER CODE
	50	58	9A	038C	651	BRB	: AND OUTPUT IT
				038E	652	OUTR8: MOVZBL	: GET CHAR TO OUTPUT
				0391	653	OUTCHAR:	: OUTPUT CHAR IN R0
				0391	654	.IF	: .IF
	5C	D5		0391	655	TSTL	: NDF, SW_PROCESS
	05	12		0393	656	BNEQ	: AP
				0395	657	MFPR	: 10\$
	51	22	DB	0395	658	MFPR	: #PRS_TXCS, R1
		04	11	0398	658	BRB	: #PRS_TXCS, R1
51	04	AC	B0	039A	659	10\$: MOVW	: 20\$
EF	51	07	E1	039E	660	20\$: BBC	: OUTCR(AP), R1
		5C	D5	03A2	661	TSTL	: #7, R1, OUTCHAR
		04	12	03A4	662	BNEQ	: AP
	23	50	DA	03A6	663	MTPR	: 30\$
			05	03A9	664	RSB	: R0, #PRS_TXDB
06	AC	50	90	03AA	665	30\$: MOVB	: RO, OUTB(AP)
				03AE	666	.IFF	: .IFF
				03AE	667	PUSHL	: RO
				03AE	668	MOVL	: SP, RO
				03AE	669	\$QIO_S	: EFN=#30, -
				03AE	670		: CHAN=TTCHAN, -
				03AE	671		: FUNC=#IOS_WRITEVBLK, -
				03AE	672		: P1=(RO), -
				03AE	673		: P2=#1
				03AE	674	POPR	: #^M<RO>
				03AE	675	.ENDC	: .ENDC
		05		03AE	676	RSB	: RSB
				03AF	677	OUTSPACE:	: AND RETURN
50	20	9A	03AF	678	MOVZBL	#32, RO	: SET CODE FOR SPACE
		DD	11	0382	679	BRB	: OUTCHAR
50	0D	9A	0384	680	CRLF: MOVZBL	#CR, RO	: AND SEND IT
		D8	10	0387	681	BSBB	: RETURN
50	0A	9A	0389	682	MOVZBL	#LF, RO	: SEND IT
		D3	11	038C	683	BRB	: LINE FEED
				038E	684		: SEND IT
				038E	685		

```

03BE 687 .SBTTL GETCHAR - GET INPUT CHARACTER ROUTINE
03BE 688
03BE 689 :
03BE 690 : GETCHAR - GET INPUT CHARACTER
03BE 691 :
03BE 692 : OUTPUT:
03BE 693 : R8 - INPUT CHARACTER
03BE 694 : R9 - BUFFER POINTER UPDATED (BUFFER IN ASCIZ FORMAT)
03BE 695 :
03BE 696
03BE 697 GETCHAR:
58 89 9A 03BE 698 MOVZBL (R9)+,R8 : GET NEXT CHARACTER
01 13 03C1 699 BEQL 10$ : READ IF NONE AVAIL
59 AC AB 9E 03C3 700 RSB :
03C4 701 10$: MOVAB INBUF-B(R11),R9 : SET ADDRESS OF INPUT BUFFER
03C8 702 .IF NDF,SW_PROCESS :
50 5C DS 03C8 703 20$: TSTL AP : CHECK FOR CONSOLE
05 13 03CA 704 BEQL 30$ : YES
50 6C B0 03CC 705 MOVW RDCR(AP),R0 : GET STATUS
03 11 03CF 706 BRB 40$ : CHECK STATUS
50 20 DB 03D1 707 30$: MFPR #PR$_RXCS,R0 : GET CONSOLE STATUS
FO 50 07 E1 03D4 708 40$: BBC #7,R0,20$ : WAIT FOR READY
5C DS 03D8 709 TSTL AP : CHECK FOR CONSOLE
58 02 AC 90 03DA 710 BEQL 50$ : YES
03 11 03DC 711 MOVB RDBUF(AP),R8 : GET CHARACTER
03E0 712 BRB 60$ : MERGE WITH COMMON
58 21 DB 03E2 713 50$: MFPR #PR$_RXDB,R8 : GET CONSOLE CHARACTER
03E2 714 .IFF : FALSE IF PROCESS VERSION
03E5 715 15$: MOVAB TERMASKD,R1 : ADDRESS OF TERMINATOR MASK DESCR
03E5 716 $Q10W_S EFN=#31,- :
03E5 717 CHAN=TTCHAN,- : INPUT DEVICE CHANNEL
03E5 718 IOSB=TTIOSB,- : IO STATUS BLOCK
03E5 719 FUNC=#<10$_READVBLK>,- :
03E5 720 P1=(R9),- : BUFFER ADDRESS
03E5 721 P2=#80,- : READ SIZE
03E5 722 P4=R1 :
03E5 723 MOVZWL TTIOSB+2,R0 : GET SIZE READ
03E5 724 MOVB TTIOSB+4,(R0)+[R9] : BUFFER TERMINATOR
03E5 725 CLRB (R9)[R0] : MARK END OF BUFFER
03E5 726 MOVL R9,R2 : POINT TO START OF STRING
03E5 727 20$: MOVZBL (R2)+,R8 : GET A CHARACTER
03E5 728 BEQL 15$ : EMPTY, READ SOME MORE
03E5 729 .ENDC :
58 80 8F 8A 03E5 730 60$: BICB #^X80,R8 : STRIP PARITY
7F 8F 58 91 03E9 731 CMPB R8,#RUBOUT : CHECK FOR RUBOUT
15 12 03ED 732 BNEQ 90$ : NO
03 6A 06 E2 03EF 733 BBSS #V RUB,(R10),70$ : SET START OF RUBOUT SEQUENCE
FF92 30 03F3 734 BSBW OUTBSL$H : OUTPUT BACK SLASH
58 79 9A 03F6 735 70$: MOVZBL -(R9),R8 : GET RUBBED OUT CHAR
04 12 03F9 736 BNEQ 80$ : SKIP INC
59 D6 03FB 737 INCL R9 : POINT AT START OF BUFFER
C9 11 03FD 738 BRB 20$ : AND GET ANOTHER
FF8C 30 03FF 739 80$: BSBW OUTR8 : OUTPUT RUBBED OUT CHAR
03 6A 06 E5 0402 740 BRB 20$ : AND GET ANOTHER
0404 741 90$: BBCC #V RUB,(R10),100$ : TERMINATE RUBOUT SEQUENCE

```

		FF7D	30	0408	742	BSBW	OUTBSLSH	:	OUTPUT BACK SLASH
03	58	06	E1	0408	743	BBC	#6,R8,110\$	:	BR IF NOT ALPHA
	58	20	8A	040F	744	BICB	#32,R8	:	SET TO UPPER CASE
				0412	745			:	
				0412	746	.IF	NDF,SW_PROCESS	:	
		FF79	30	0412	747	BSBW	OUTR8	:	ECHO CHARACTER
				0415	748	.ENDC		:	
FD63	CF	89	58	90	0415	MOVB	R8,(R9)+	:	BUFFER NEW CHAR
		08	58	3A	0418	LOCC	R8,#NTERM,TERM	:	CHECK FOR TERMINATOR
				13	041E	BEQL	20\$	:	NOT A TERMINATOR
		58	0D	91	0420	CMPB	#CR,R8	:	IS CHAR = RETURN
				12	0423	BNEQ	120\$	:	NO
		FF8C	30	0425	754	BSBW	CRLF	:	YES, SEND CR/LF
				69	94	CLRB	(R9)	:	MARK END OF BUFFER
59		AC AB	9E	042A	756	MOVAB	INBUF-B(R11),R9	:	RESTORE BUFFER BASE
		FF8D	31	042E	757	BRW	GETCHAR	:	AND TRY AGAIN

```

0431 759 .SBTTL PLUS/MINUS OPERATORS
0431 760 :
0431 761 : PLUS/MINUS OPERATORS
0431 762 :
0431 763 BLANK: : SAME AS PLUS
0431 764 OPERATOR: :
FF AB 50 FDEB 30 0431 765 BSBW ENDEXPR : END EXPR
83 0434 766 SUBB3 #OPERBAS,R0,OPER-B(R11) : SET OPERATOR
05 0439 767 RSB : RETURN
043A 768 :
043A 769 : MONADIC MINUS - NEGATE
043A 770 :
56 D5 043A 771 NEGATE: TSTL R6 : TEST ACCUMULATOR
03 13 043C 772 BEQL 5$ : EMPTY
FDDE 30 043E 773 BSBW ENDEXPR : OTHERWISE PERFORM OPERATION
6A 80 8F 8C 0441 774 5$: XORB #<1@V_NEGATE>,(R10) : TOGGLE NEGATE FLAG
05 0445 775 10$: RSB : AND RETURN
0446 776
0446 777

```



```

0446 779 .SBTTL TAB - INDIRECT DISPLAY
0446 780 :
0446 781 :
0446 782 :
0446 783 TAB:
50 6A 01 04 AB DO 0446 783 TAB:
6A 01 1F 50 EF 044A 784 : GO INDIRECT
6A 01 1F 50 FO 044F 785 : GET PROCESSOR REGISTER MODE
OF 11 0454 786 : AND COPY TO SEMI-PERMANENT FLAG
0456 787 : AND DISPLAY IT
0456 788 :
0456 789 :
0456 790 :
0456 791 :
0456 792 ESCAP:
51 01 DO 0456 793 :
6A D5 0459 794 : ASSUME UNIT INCREMENT
05 19 045B 795 : CHECK FOR PROCESSOR REGISTER
51 51 FE AB 9C 045D 796 : YES, USE UNIT INCREMENT
68 51 C2 0462 797 10$: : FORM INCREMENT
FEBF 31 0465 798 LOCP: : AND SUBTRACT FROM DOT
: PROMPT WITH CONTENT
MOVL #1,R1 :
TSTL (R10) :
BLSS 10$ :
ROTL CURTYPE-B(R11),R1,R1 :
SUBL R1,CURDOT-B(R11) :
BRW LOCPROMPT :

```

```

0468 800 .SBTTL EQUALS - DISPLAY VALUE
0468 801 :
0468 802 :
0468 803 :
0468 804 EQUALS:
0468 805 .ENABL LSB
0468 806 BSBW ENDFIELD :
0468 807 BBC #V F1,(R10),10$ :
046F 808 EQL1: MOVL F1=B(R11),QUAN-B(R11) :
0474 809 10$: BSBW OUTPUT :
0477 810 : BRB RESET :
0477 811 : .DSABL LSB :
0477 812 :
0477 813 :
0477 814 :
0477 815 :
0477 816 :
6A 00FFFF80 8F CA 0477 817 RESET: BICL #*X0FFFF80,(R10) :
FC AB 94 047E 818 CLRB FCTR-B(R11) :
56 7C 0481 819 CLRQ R6 :
05 0483 820 RSB :

```

```

:
: TERMINATE FIELD
: IGNORE IF FIELD BLANK
: SET QUANTITY
: OUTPUT IT
: RESET SCANNER
:
:
: CLEAR FIELD AND NEGATE FLAGS
: CLEAR FIELD COUNTER
: RESET ACCUMULATORS
: RETURN

```

```

0484 822 .SBTTL SEMI - SECONDARY COMMAND SET
0484 823 :
0484 824 : SEMI
0484 825 :
0484 826 :
0484 827 SECOND:
58 0484 828 .ASCII /X/ : X REGISTER SET/DISPLAY
50 0485 829 .ASCII /P/ : P - PROCEED
4D 0486 830 .ASCII /M/ : M - SET MODIFY FLAG
49 0487 831 .ASCII /I/ : I - PROGRAM COUNTER
47 0488 832 .ASCII /G/ : G - GO, START
45 0489 833 .ASCII /E/ : E - EXECUTE STRING
42 048A 834 .ASCII /B/ : B - SET/CLR BREAKPOINT
00000007 048B 835 NSEC=-SECOND : NUMBER OF SECONDARY COMMANDS
048B 836
048B 837 SEMI:
6A 01 8A 048B 838 BICB #<1@V OPEN>,(R10) : CLEAR OPEN FLAG
FE0D 30 048E 839 BSBW ENDFIELD : TERMINATE FIELD
FF2A 30 0491 840 BSBW GETCHAR : GET SECONDARY COMMAND CHAR
EB AF 07 58 3A 0494 841 LOCC R8,#NSEC,SECOND : LOCATE SECONDARY COMMAND
0499 842 10$: CASE RO,LIMIT=#1,<- : SWITCH ON TYPE
0499 843 BRKPOINT,- : SET BREAKPOINT
0499 844 EXECUTE,- : EXECUTE STRING
0499 845 GO,- : SEMI-G, GO
0499 846 PROGCTR,- : SEMI-I, INSTRUCTION CONTER
0499 847 MFYFLGS,- : SEMI-M, MODIFY FLAG
0499 848 PROCED,- : SEMI-P, PROCEED
0499 849 XSET,- : SET XREGISTER
0499 850 >
06' 01 50 AF 0499 CASEW RO,#1,S^#<<30011$-30010$>/2>-1
049D 30010$: .SIGNED_WORD BRKPOINT-30010$
003A' 049D .SIGNED_WORD EXECUTE-30010$
037A' 049F .SIGNED_WORD GO-30010$
00D8' 04A1 .SIGNED_WORD PROGCTR-30010$
0105' 04A3 .SIGNED_WORD MFYFLGS-30010$
00EC' 04A5 .SIGNED_WORD PROCED-30010$
00E1' 04A7 .SIGNED_WORD XSET-30010$
0133' 04A9
04AB 30011$:
FCFO 31 04AB 851 ERR2: BRW ERROR : ERROR

```

```

04AE 853 .SBTTL LEFT BRACKET - MODE SELECTION
04AE 854 :
04AE 855 :
04AE 856 : LEFT BRACKET
04AE 857 :
04AE 858 MODES: : MODE CHARACTER LIST
43 04AE 859 .ASCII /C/ : CHARACTER
4C 04AF 860 .ASCII /L/ : LONG, HEX
57 04B0 861 .ASCII /W/ : WORD, HEX
42 04B1 862 .ASCII /B/ : BYTE, HEX
00000004 04B2 863 NMODES=.-MODES : NUMBER OF MODE CHARACTERS
04B2 864
04B2 865
04B2 866 LBRACKET: : MODE SELECTION
F4 AF 04 FF09 30 04B2 867 BSBW GETCHAR : GET MODE CHAR
09 50 02 EF 13 04B5 868 LOCC R8,#NMODES,MODES : CONVERT TO INDEX
FE AB 50 01 E0 04BC 869 BEQL ERR2 : NOT FOUND, ERROR
6A 02 8A 04C0 870 BBS #2,R0,10$ : CHECK FOR 'C'
05 04C8 871 SUBB3 #1,R0,CURTYPE-B(R11) : SET MODE
6A 02 88 04C5 872 BICB #<1@V_ASCII>,(R10) : CLEAR CHAR MODE
05 04C9 873 RSB : RETURN
05 04CC 874 10$: BISB #<1@V_ASCII>,(R10) : SET CHARACTER MODE
05 04CC 875 RSB

```

					04CD	877		.SBTTL	SINGLE STEP		
					04CD	878	:				
					04CD	879	:	STEP			
					04CD	880	:				
6A	02	03	01	F0	04CD	881	STEP:	INSV	#1,#V TBIT,#2,(R10)	:	CLR V_ATBRK, SET V_TBIT
	00	6A	0F	E5	04D2	882		BBCC	#V_PMODE,(R10),20\$	:	CLEAR-PROCESSOR REGISTER DISPLAY MODE
				04	04D6	883	20\$:	RET		:	AND RETURN

```

04D7 885 .SBTTL BRKPOINT - SET/CLEAR BREAKPOINTS
04D7 886 :
04D7 887 : BRKPOINT
04D7 888 :
04D7 889 BRKPOINT:
58 6A 08 E1 04D7 890 BBC #V F1,(R10),SHOBRK : DISPLAY BREAKPOINTS
12 6A 09 E0 04DB 891 BBS #V F2,(R10),20$ : YES, IT WAS SPECIFIED
52 01 D0 04DF 892 MOVL #1,R2 : INIT INDEX
FBD1 CF42 D5 04E2 893 10$: TSTL BRKADR[R2] : FIND FREE SLOT
13 13 04E7 894 BEQL 30$ : YES, GOT ONE
FFF3 52 01 08 F1 04E9 895 ACBL #NBRK,#1,R2,10$ : CHECK THEM ALL
52 E4 AB BA 11 04EF 896 BRB ERR2 : ERROR
52 08 D0 04F1 897 20$: MOVL F2-B(R11),R2 : GET BRKPOINT NUMBER
EB 13 04F5 898 BEQL 10$ : NULL FIELD, SCAN FOR SLOT
52 08 D1 04F7 899 CML #NBRK,R2 : CHECK FOR LEGAL
AF 19 04FA 900 BLSS ERR2 : OUT OF RANGE
FBDF CF42 D4 04FC 901 30$: CLRL BRKDSP[R2] : CLEAR DISPLAY
FBFA CF42 D4 0501 902 CLRL BRKCOM[R2] : CLEAR COMMAND ADDRESS
50 E0 AB D0 0506 903 MOVL F1-B(R11),R0 : GET BREAKPOINT ADDRESS
03 13 050A 904 BEQL 35$ : ALLOW CLEAR OF BREAKPOINT
050C 905 .IF DF,SW PROCESS
050C 906 PUSH #M<R0,R1,R2,R3,R4,R5,R6> : ; SAVE REGISTERS FOR PROTECTION CHAN
050C 907 MOVL R0,R5 : SET START ADDRESS
050C 908 MOVL R0,R6 : AND END ADDRESS
050C 909 BSBW SETWRT : SET PAGE WRITABLE
050C 910 MOVL (SP),R0 : RESTORE BPT ADDRESS
60 60 90 050C 911 .ENDC
050F 912 MOVB (R0),(R0) : TEST WRITABILITY OF ADDRESS
050F 913 .IF DF,SW PROCESS
050F 914 BSBW REPROT : RESTORE PROTECTION
050F 915 POPR #M<R0,R1,R2,R3,R4,R5,R6> : ; AND REGISTERS
050F 916 .ENDC
OC 6A 0A E1 050F 917 35$: BBC #V F3,(R10),40$ : DISPLAY SPECIFIED?
FBC6 CF42 E8 AB D0 0513 918 MOVL F3-B(R11),BRKDSP[R2] : SET DISPLAY START
03 13 051A 919 BEQL 40$ : SKIP TEST IF NULL
E8 BB D5 051C 920 TSTL @F3-B(R11) : CHECK READABILITY
07 6A 08 E1 051F 921 40$: BBC #V F4,(R10),45$ : SKIP IF NO COMMAND ADDRESS
FBD6 CF42 EC AB D0 0523 922 MOVL F4-B(R11),BRKCOM[R2] : SET COMMAND STRING
FB88 CF42 50 D0 052A 923 45$: MOVL R0,BRKADR[R2] : SAVE BREAKPOINT ADDRESS
FF44 31 0530 924 BRW RESET : RESET SCANNER AND RETURN
0533 925 :
0533 926 : SHOBRK
0533 927 :
0533 928 SHOBRK:
58 55 01 D0 0533 929 MOVL #1,R5 : INIT INDEX FOR LOOP
FB7D CF45 D0 0536 930 10$: MOVL BRKADR[R5],R8 : GET BREAKPOINT ADDRESS
2E 13 053C 931 BEQL 20$ : SKIP IF NULL
53 55 D0 053E 932 MOVL R5,R3 : BREAKPOINT NUMBER
FE70 30 0541 933 BSBW CRLF : NEW LINE
FE16 30 0544 934 BSBW OUTDIGIT : BPT NUMBER
FF65 30 0547 935 BSBW OUTSPACE : SPACE
53 58 D0 054A 936 MOVL R8,R3 : ADDRESS OF BPT
FE11 30 054D 937 BSBW OUTLONG : OUTPUT ADDRESS
F5C 30 0550 938 BSBW OUTSPACE : SPACE OVER
53 FB88 CF45 D0 0553 939 MOVL BRKDSP[R5],R3 : GET DISPLAY START
03 13 0559 940 BEQL 15$ : NONE
FE03 30 055B 941 BSBW OUTLONG : OUTPUT DISPLAY START

```

53	FB9D	CF45	D0	055E	942	15\$:	MOVL	BRKCOM[R5],R3	:	GET COMMAND STRING ADDRESS
		06	13	0564	943		BEQL	20\$	:	NONE
		FE46	30	0566	944		BSBW	OUTSPACE	:	SPACE ANOTHER
		FDF5	30	0569	945		BSBW	OUTLONG	:	AND OUTPUT A LONGWORD
FFC4	55	01	08	F1	056C	946	20\$:	ACBL	:	DO THEM ALL
		FE3F	31	0572	947		BRW	CRLF	:	AND EXIT THROUGH CRLF

```
0575 949 .SBTTL GO - START EXECUTION AT SPECIFIED LOCATION
0575 950 :
0575 951 :
0575 952 :
0575 953 GO: BBC #V_F1,(R10),PROCEED : JUST PROCEED IF NO VALUE
54 AB 05 6A 08 E1 0579 954 MOVL F1=B(R11),SAVPC-B(R11) : SET NEW PC
EO AB D0 057E 955 : BRW PROCEED : FALL INTO PROCEED
057E 956 :
057E 957 : PROCEED
057E 958 :
04 057E 959 PROCED: :
057E 960 RET : RETURN
```



```

057F 962 .SBTTL SEMI-I, PC VALUE
057F 963 :
057F 964 : SEMI-I
057F 965 :
057F 966 COLON: BSBW ENDEXPR : TERMINATE EXPRESSION
F8 AB FC9D 30 D0 0582 967 MOVL R7,PID-B(R11) : SET PID FOR PROCESS
57 7C 0586 968 CLRQ R6 : RESET ACCUMULATORS
56 05 0588 969 RSB :
0589 970 :
51 F4 AB DE 0589 971 MFYFLGS:MOVAL MFYFLG-B(R11),R1 : SET MODIFY FLAG ADDRESS
17 11 058D 972 BRB VALUE : SET/GET VALUE
51 51 6B DE 058F 973 DOT: MOVAL CURDOT-B(R11),R1 : SET ADDRESS OF DOT
18 6A 1F E1 0592 974 BBC #V_PREG,(R10),VALR : WAS IT PROCESSOR REGISTER?
14 6A 0F E2 0596 975 BBSS #V_PMODE,(R10),VALR : YES, SET PROCESSOR REGISTER MODE
12 11 059A 976 BRB VALR : READ VALUE
51 04 AB DE 059C 977 QUANT: MOVAL QUAN-B(R11),R1 : SET QUANTITY ADDRESS
0C 11 05A0 978 BRB VALR : READ VALUE
05A2 979 PROGCTR: :
51 54 AB DE 05A2 980 MOVAL SAVPC-B(R11),R1 : SET PC ADDRESS
04 6A 08 E1 05A6 981 VALUE: BBC #V_F1,(R10),VALR : SKIP IF NO VALUE
61 E0 AB D0 05AA 982 MOVL F1-B(R11),(R1) : SET NEW VALUE FOR PC
56 61 D0 05AE 983 VALR: MOVL (R1),R6 ; AND GET VALUE
FC^2 31 05B1 984 VALI: BRW INFLD : SET FIELD IN PROGRESS
05B4 985 REGISTER: :
55 18 AB DE 05B4 986 MOVAL SAVREG-B(R11),R5 : SET BASE OF REGISTER AREA
02 10 05B8 987 BSBB REGCOM : FETCH ADDRESS
F5 11 05BA 988 BRB VALI : AND USE IT
FBA3 CF 10 FDFD 30 05BC 989 REGCOM: BSBW GETCHAR : GET SECOND CHAR
58 3A 05BF 990 LOCC R8,#16,PRIMARY : TRANSLATE TO HEX
05C5 991 .IF DF_SW_PROCESS : FOR PROCESS VERSION
05C5 992 BNEQ 10$ : LEGAL HEX DIGIT
05C5 993 CMPW #^A/XI/,-2(R9) : CHECK FOR EXIT COMMAND
05C5 994 BNEQ ERR3 : NO, ERROR
05C5 995 $EXIT_S EXITCODE : YES EXIT
05C5 996 .IFF :
43 13 05C5 997 BEQL ERR3 : ERROR, NOT HEX
05C7 998 .ENDC :
05C7 999 10$: :
50 10 50 C3 05C7 1000 SUBL3 R0,#16,R0 : INVERT
56 6540 DE 05CB 1001 MOVAL (R5)[R0],R6 : ACCUMULATE
05 05CF 1002 RSB : RETURN
05D0 1003 :
51 E4 AB 36 6A 09 E1 05D0 1004 XSET: BBC #V_F2,(R10),ERR3 : ERROR IF NOT TWO FIELDS
04 00 EF 05D4 1005 EXTZV #0,#4,F2-B(R11),R1 : GET REGISTER NUMBER
51 FB45 CF41 DE 05DA 1006 MOVAL XREGV[R1],R1 : AND COMPUTE REGISTER ADDRESS
C4 11 05E0 1007 BRB VALUE : PROCESS VALUE
05E2 1008 XREG: : X-REGISTER VALUE
55 FB3E CF DE 05E2 1009 MOVAL XREGV,R5 : SET ADDRESS OF REGISTER VECTOR
D3 10 05E7 1010 BSBB REGCOM : ADDRESS TO R6
56 66 D0 05E9 1011 MOVL (R6),R6 : GET VALUE
C3 11 05EC 1012 BRB VALI : AND NOTE INPUT IN FIELD
05EE 1013 .ALIGN LONG : LONGWORD ALIGN EXCEPTION ROUTINES
05F0 1014 XDFLACV: : ACCESS VIOLATION HANDLER
05F0 1015 MCHK: : MACHINE CHECK
05F0 1016 .IF NDF_SW_PROCESS :
5C D5 05F0 1017 TSTL AP : CHECK FOR SIMULATOR
16 12 05F2 1018 BNEQ ERR3 : YES, SKIP RESET

```

			05F4	1019					
			05F4	1020					
			05F4	1021	CPUDISP	<CLR_780,-			: *DISPATCH ON CPU TYPE*
			05F4	1022		CLR_750,-			: :
			05F4	1023		CLR_730>			: :
02'	01	00000000'	GF	8F	05F4	CASEB	G^EXE\$GB_CPUTYPE,#PRS_SID_TYP780,S^#<<30013\$-30012\$>/2>-1		
					05FC			30012\$:	
		0006'			05FC		.SIGNED_WORD CLR_78C-30012\$		
		000B'			05FE		.SIGNED_WORD CLR_750-30012\$		
		000B'			0600		.SIGNED_WORD CLR_730-30012\$		
					0602			30013\$:	
					0602				
					0602		CLR_780:		: FOR 11/780:
30	00	DA			0602		MTPR #0,#PRS_SBIFS		: CLEAR SBI FAULT
	03	11			0605		BRB CLR_END		: ERROR CLEARED
					0607				: :
					0607		CLR_730:		: FOR 11/730:
					0607		CLR_750:		: FOR 11/750:
26	0F	DA			0607		MTPR #^XF,#PRS_MCESR		: SET 1 TO CLEAR MCHECK ERROR SUMMARY
					060A				: :
					060A		CLR_END:		: *END OF CPU-DEPENDENT CODE*
					060A				: :
					060A				: :
					060A		.ENDC		: :
					060A		10\$:		: :
FB91	31				060A		ERR3: BRW ERROR		: AND DECLARE ERROR
					060D				: :

```

060D 1041 .SBTTL REGISTER SAVE AND RESTORE
060D 1042
060D 1043 :
060D 1044 : SAVE - SAVE TARGET PEGISTERS, PC, PSL
060D 1045 :
060D 1046 : SAVE:
060D 1047 :
060D 1048 :
12 1F DA 060D .IF NDF,SW_PROCESS :
FA5B CF 50 7D 0610 #31 : DISABLE
51 FA5F CF 9E 0610 MTPR #31,S^#PRS_IPL :
0610 1049 : JSB INISWRITABLE : MAKE THE SYSTEM WRITABLE
0610 1050 : MOVQ R0,SAVREG : SAVE R0,R1
0615 1051 : MOVAB SAVR2,R1 : SETUP BASE FOR REMAINING REGS
061A 1052 : .IFF : FALSE IF PROCESS VERSION
061A 1053 : $SETAST_S #0 : DISABLE ASTS
061A 1054 : PUSHAB -(R0) : SAVE ENABLE VALUE-1
061A 1055 : MOVPSL R1 : GET CURRENT PSL
061A 1056 : EXTZV #PSL$V_CURMOD,#PSL$S_CURMOD,R1,R1 : ISOLATE CURRENT MODE
061A 1057 : MULW #CONTEXTSZ,R1 : COMPUTE OFFSET TO PROPER CONTEXT AREA
061A 1058 : MOVAB SAVREG[R1],R1 : FORM ADDRESS OF REGISTER SAVE
061A 1059 : MOVL 8(AP),R0 : GET POINTER TO MECHANISM
061A 1060 : MOVQ 12(R0),(R1)+ : SAVE R0,R1
061A 1061 : .ENDC
81 52 7D 061A 1062 : MOVQ R2,(R1)+ : SAVE R2,R3
81 54 7D 061D 1063 : MOVQ R4,(R1)+ : SAVE R4,R5
81 56 7D 0620 1064 : MOVQ R6,(R1)+ : SAVE R6,R7
81 58 7D 0623 1065 : MOVQ R8,(R1)+ : SAVE R8,R9
81 5A 7D 0626 1066 : MOVQ R10,(R1)+ : SAVE R10,R11
0629 1067 : .IF NDF,SW_PROCESS :
81 81 5C 7D 0629 1068 : MOVQ AP,(R1)+ : SAVE AP,FP
81 0C AE 9E 062C 1069 : MOVAB 12(SP),(R1)+ : ASSUME KERNEL STACK
81 04 AE 7D 0630 1070 : MOVQ 4(SP),(R1)+ : SAVE PC,PSL
0634 1071 : .IFF
0634 1072 : MOVQ 8(FP),(R1)+ : SAVE AP,FP
0634 1073 : SUBL3 #1,@4(AP),R0 : GET NUMBER OF ARGS IN SIGNAL
0634 1074 : MOVAL @4(AP)[R0],R0 : POINT TO PC,PSL
0634 1075 : MOVAL 8(R0),(R1)+ : COMPUTE SP
0634 1076 : MOVQ (R0),(R1)+ : SAVE PC,PSL
0634 1077 : .ENDC
0634 1078 : .IF NDF,SW_PROCESS :
81 22 DB 0634 1079 : MFPR #PRS_TXCS,(R1)+ : SAVE CONSOLE TRANSMIT STATUS
81 20 DB 0637 1080 : MFPR #PRS_RXCS,(R1)+ : SAVE CONSOLE RECVR STATUS
5C D4 063A 1081 : CLRL AP : ZAP DEVICE ADDRESS BASE
063C 1082 : .ENDC
22 00 DA 063C 1083 : .IF NDF,SW_PROCESS :
20 00 DA 063F 1084 : MTPR #0,#PRS_TXCS : CLEAR INTERRUPT ENABLE
0642 1085 : MTPR #0,#PRS_RXCS : FOR BOTH TRANSMIT AND RECEIVE
0642 1086 : .ENDC
SB FA12 CF 9E 0642 1087 : .IF NDF,SW_PROCESS :
0642 1088 20$: MOVAB B,R11 : AND DATA BASE ADDRESS
0647 1089 : .IFF : FALSE FOR PROCESS VERSION
0647 1090 : MOVAB W^<B-<SAVPSL+4>>(R1),R11 : SET BASE OF CONTEXT AREA
0647 1091 : MOVL (SP)+,ASTEN-B(R11) : SAVE AST ENABLE
0647 1092 : .ENDC
5A DC AB 9E 0647 1093 : MOVAB STATUS-B(R11),R10 : SET STATUS BASE
59 AC AB 9E 064B 1094 : MOVAB INBUF-B(R11),R9 : POINT TO INPUT BUFFER

```

```

69 94 064F 1095      CLR      (R9)                ; MAKE BUFFER EMPTY
      0651 1096      .IF      NDF,SW_PROCESS                ;
      0651 1097      BSBW     GET$SCB-                   ; GET BASE OF SCB
FBOA CF 0084 30 0651 1097      MOVL     4(R0),MCHKSAV        ; SAVE ORIGINAL MCHK VECTOR
04 AO 04 AO 93 AF 9E 0654 1098      MOVAB   MCHK,4(R0)          ; SET TO XDELTA VECTOR
20 AO 04 AO 8E AF 9E 065A 1099      MOVAB   XDELACV,^X20(R0)      ; SET ACCESS VIOLATION VECTOR
24 AO 04 AO 89 AF 9E 065F 1100      MOVAB   XDELACV,^X24(R0)      ; SET PG FAULT VECTOR
18 AO 04 AO 84 AF 9E 0664 1101      MOVAB   XDELACV,^X18(R0)     ; SET RESERVED OPERAND HANDLER
50 08 AE 02 18 EF 066E 1103      EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,8(SP),R0 ; GET MODE
      0674 1104      BEQL     30$                          ; CORRECT ALREADY IF KERNEL
      50 00 C0 0676 1105      ADDL     #PRS_KSP,R0                ; COMPUTE PROCESSOR REGISTER
      0679 1106      MFPR     R0,SAVSP-B(R11)             ; AND SAVE CORRECT SP
      50 AB 50 DB 0679 1106      MFPR     R0,SAVSP-B(R11)
      067D 1107      .ENDC
      FDF7 31 067D 1108 30$: BRW     RESET                ; RESET SCANNER
      0680 1109
      0680 1110      ;
      0680 1111      ; RESTORE - RESTORE TARGET REGISTERS
      0680 1112      ;
      0680 1113      RESTORE: ; RESTORE EVERYTHING
04 AE 54 AB 7D 0680 1114      .IF      NDF,SW_PROCESS                ;
      0680 1115      MOVQ     SAVPC-B(R11),4(SP)          ; SET PC,PSL
      0685 1116      .IFF
      0685 1117      SUBL3   #1,@4(AP),R0                ; FALSE IF PROCESS
      0685 1118      MOVAL   @4(AP)[R0],R0                ; GET SIGNAL ARG COUNT
      0685 1119      MOVQ     SAVPC-B(R11),(R0)           ; COMPUTE ADDRESS OF PC,PSL
      0685 1120      .ENDC ; STORE UPDATED PC,PSL
      0685 1121      RESTORR: ; RESTORE REGISTERS ONLY
      0685 1122      .IF      NDF,SW_PROCESS                ;
      0685 1123      BSBB     GET$SCB-                   ; GET BASE OF SCB
20 AO 00000000'EF 9E 0687 1124      MOVAB   EXESACVIOLAT,^X20(R0)      ; RESTORE ACCESS VECTOR
24 AO 00000000'EF 9E 068F 1125      MOVAB   MMG$PAGEFAULT,^X24(R0) ; AND PAGE FAULT VECTOR
04 AO FAC9 CF DO 0697 1126      MOVL     MCHKSAV,4(R0)          ; RESTORE MACHINE CHECK VECTOR
18 AO 00000000'EF 9E 069D 1127      MOVAB   EXESROPRAND,^X18(R0) ; RESTORE RESERVED OPERAND VECTOR
      5C B5 06A5 1128      TSTW     AP                    ; CHECK FOR CONSOLE
      0A 12 06A7 1129      BNEQ     10$                          ; NO, OTHER DEVICE
      22 5C AB DA 06A9 1130      MTPR     SAVOCR-B(R11),#PRS_TXCS ; RESTORE INITIAL TX STATUS
      20 60 AB DA 06AD 1131      MTPR     SAVRXCS-B(R11),#PRS_RXCS ; AND INITIAL RECEIVER STATE
04 AC 5C AB B0 06B1 1132      BRB     20$                          ; MERGE WITH COMMON CODE
      6C 5E AB B0 06B3 1133 10$: MOVW   SAVOCR-B(R11),OUTCR(AP) ; RESTORE OUTPUT CSR
      06B8 1134      MOVW   SAVRCR-B(R11),RDCR(AP) ; AND INPUT CSR CONTENT
      06BC 1135      .IFF
      06BC 1136      PUSHL   ASTEN-B(R11)                ; SAVE AST ENABLE
      06BC 1137      .ENDC
      51 20 AB 9E 06BC 1138 20$: MOVAB   SAVR2-B(R11),R1 ; SET BASE FOR RESTORE
      52 81 7D 06C0 1139      MOVQ     (R1)+,R2            ; RESTORE R2,R3
      54 81 7D 06C3 1140      MOVQ     (R1)+,R4            ; RESTORE R4,R5
      56 81 7D 06C6 1141      MOVQ     (R1)+,R6            ; RESTORE R6,R7
      58 81 7D 06C9 1142      MOVQ     (R1)+,R8            ; RESTORE R8,R9
      5A 81 7D 06CC 1143      MOVQ     (R1)+,R10           ; RESTORE R10,R11
      06CF 1144      .IF      NDF,SW_PROCESS                ;
      50 5C 81 7D 06CF 1145      MOVQ     (R1)+,AP            ; RESTORE AP,FP
      F99A CF 7D 06D2 1146      MOVQ     SAVREG,R0          ; RESTORE R0,R1
      06D7 1147      .IFF ; FALSE IF PROCESS VERSION
      06D7 1148      MOVQ     (R1)+,8(FP)                ; SET NEW VALUES FOR AP,FP
      06D7 1149      MOVL     8(AP),R0                    ; GET MECHANISM POINTER
      06D7 1150      MOVQ     <SAVREG-SAVSP>(R1),12(R0) ; STORE UPDATED R0,R1

```

```
06D7 1151      MOVPSL R1      : GET CURRENT PSL
06D7 1152      EXTZV  #PSL$V_CURMOD,#PSL$$_CURMOD,R1,R1 : GET CURRENT MODE
06D7 1153      BBCC   R1,DBG$ACTIVE,30$      : CLEAR ACTIVE BIT FOR MODE
06D7 1154 30$:  TSTL   (SP)+      : CHECK FOR AST ENABLE
06D7 1155      BEQL   35$      : NO
06D7 1156      $SETAST_S      #1      : RE- ENABLE AST RECOGNITION
06D7 1157
06D7 1158 35$:  .ENDC
06D7 1159      .IF      NDF,SW_PROCESS
06D7 1160      JSB     INI$RDONLY      : REPROTECT THE SYSTEM CODE
06D7 1161 :
06D7 1162      .ENDC
05 06D7 1163      RSB      : AND RETURN
```

```

06D8 1166          .SBTTL GET SCB ADDRESS
06D8 1167
06D8 1168          :
06D8 1169          : SUBROUTINE GETSCB IS CALLED TO GET THE PHYSICAL OR VIRTUAL
06D8 1170          : ADDRESS OF THE CURRENT SCB.
06D8 1171          :
06D8 1172          : INPUTS:          NONE
06D8 1173          :
06D8 1174          : OUTPUTS:         RO = SCB ADDRESS
06D8 1175          :                   OTHER REGISTERS PRESERVED
06D8 1176          :
06D8 1177          :
06D8 1178          : .IF          NDF,SW PROCESS          : NOT FOR PROCESS VERSION
06D8 1179 GETSCB: MFPR #PR$_MAPEN,RO      : GET MAPPING STATUS
06D8 1180          : MFPR #PR$_MAPEN,RO
06DD 1181          : BNEQ          10$          : BRANCH IF MAPPING ENABLED
06DD 1181          : MFPR          #PR$_SCBB,RO : ELSE GET PHY ADDR OF SCB
06DD 1182          : BRB          20$          : JOIN COMMON RETURN
50 00000000'EF DE 06E2 1183 10$: MOVAL SCB$AL_BASE,RO : IF MAPPING ENABLED, GET SCB VA
05 06E9 1184 20$: RSB          : RETURN
06EA 1185          : .ENDC          :

```

```

00 20 54 41 20 4B 52 42 20
      FF16 30
      00D3 30
      53 D5
      10 12
      FF84 30
      7E 06 AE 9A
      12 8E DA
      00000000'EF 17
      6A 18 88
      38 58 AB 04 E0
      55 53 D0
      FC95 30
      FC3B 30
      54 C5 AF 9E
      FC55 30
      53 54 AB D0
      FC31 30
      FC7C 30
      51 F9A8 CF45 D0
      06 13
      6B 51 D0
      FBB6 30
      51 F9BA CF45 D0
      03 13
      59 51 D0
      FA49 CF 6C FA
      09 6A 03 E5
      00 58 AB 04 E2
      075C 1239 40$:
      075C 1240
      075C 1241
      075C 1242

      .SBTTL BPT TRAP HANDLER
      :
      HANDLE BREAKPOINT TRAPS
      BMSG: .ASCIZ / BRK AT / : BREAK POINT MESSAGE
      .ALIGN LONG : LONGWORD ALIGNMENT
      .IF NDF,SW_PROCESS : EXEC VERSION
      XDELBPT: : XDELTA BPT ENTRY
      .IFF :
      XDELBPT: : DELTA BPT ENTRY
      .ENDC
      BSBW SAVE : SAVE REGS AND DISABLE
      BSBW GETBPTX : GET INDEX OF BPT
      TSTL R3 : CHECK FOR MATCH
      BNEQ 10$ : YES, FOUND IT
      BSBW RESTORR : RESTORE REGISTERS ONLY
      .IF NDF,SW_PROCESS :
      MOVZBL 6(SP),-(SP) : GET IPL
      ENBINT : ENABLE
      JMP MTPR (SP)+,S^#PR$ _IPL : AND HANDLE NORMALLY
      .IFF EXESBREAK : FALSE IF PROCESS VERSION
      :
      ***** UNEXPECTED BREAKPOINT *****
      CLRL RO : RETURN FALSE
      RET :
      .ENDC
      BSBW #<<1@V_TBIT>!<1@V_ATBRK>>, (R10) ; SET STATUS
      10$:
      30$:
      BSBW UNBRK : RESTORE OPCODES
      BBS #PSL$V_TBIT,SAVPSL-B(R11) : PROCEED ; PROCEED IF BPT AND TBIT
      MOVL R3,R5 : SAVE BPT NUMBER
      BSBW CRLF : OUTPUT CR/LF PAIR
      BSBW OUTDIGIT : OUTPUT BPT NUMBER
      MOVAB BMSG,R4 : MSG ADDRESS
      BSBW OUTZSTRING : OUTPUT ASCIIZ
      MOVL SAVPC-B(R11),R3 : OUTPUT PC
      BSBW OUTLONG : OUTPUT HEX LONGWORD
      BSBW OUTSPACE : SEND SPACE
      MOVL BRKDSP[R5],R1 : GET ADDRESS TO DISPLAY
      BEQL 40$ : NONE
      MOVL R1,CURDOT-B(R11) : SET AS CURRENT DOT
      BSBW LOC PROMPT : AND DISPLAY
      MOVL BRKCOM[R5],R1 : GET COMMAND STRING ADDRESS
      BEQL GETCMD : NONE GET COMMAND
      MOVL R1,R9 : SET TO SCAN STORED COMMAND
      GETCMD: : GET COMMANDS
      :
      CALLG (AP),DCOM : PERFORM DEBUG COMMANDS
      PROCEED: : PROCEED
      BSBW SETBRK : SET BREAKPOINTS
      BBCC #V_TBIT,(R10),50$ : TEST AND CLR TRACE FLAG
      BBSS #PSL$V_TBIT,SAVPSL-B(R11),40$ : SET TBIT
      :
      .IF DF,SW_PROCESS : FOR PROCESS VERSION
      CMPB #2,@SAVPC-B(R11) : CHECK FOR REI OPCODE
      BNEQ 45$ : NO, NOTHING SPECIAL

```

			075C	1243	EXTZV	#PSLSV_CURMOD,#PSLSS_CURMOD,SAVPSL-B(R11),R0	: GET NEW MODE	
			075C	1244	MULW	#CONTEXTSZ,R0	: SCALE BY PER MODE CONTEXT AREA SIZE	
			075C	1245	MOVAB	STATUS-B(R0),R10	: POINT TO NEW FLAGS	
			075C	1246	.ENDC			
00 6A	05	E2	075C	1247	45\$:	BBSS	#V_TBITOK,(R10),50\$	: SET TBIT EXPECTED
	FF1D	30	0760	1248	50\$:	BSBW	RESTORE	: RESTORE EVERYTHING
			0763	1249		.IF	NDF,SW_PROCESS	
		02	0763	1250		REI		: AND RETURN
			0764	1251		.IFF	#1,R0	: FALSE IF PROCESS VERSION
			0764	1252		MOVL		: RETURN TRUE
			0764	1253		RET		
			0764	1254		.ENDC		
			0764	1255				



```

0764 1257 .SBTTL TBIT EXCEPTION HANDLER
0764 1258 :
0764 1259 : HANLER FOR TBIT EXCEPTION
0764 1260 :
0764 1261 .ALIGN LONG : LONGWORD ALIGNED
0764 1262 .IF NDF,SW_PROCESS :
0764 1263 XDELTBIT:: : XDELTA TBIT HANDLER
0764 1264 .IFF :
0764 1265 XDELTBIT: :
0764 1266 .ENDC :
10 6A FE A6 30 0764 1267 BSBW SAVE : SAVE AND DISABLE
0764 1268 BBSC #V TBITOK,(R10),XDELDBG : BR IF TBIT EXPECTED
0764 1269 BSBW RESTORR : RESTORE REGISTERS
0764 1270 .IF NDF,SW_PROCESS :
0764 1271 MOVZBL 6(SP),=(SP) : GET IPL FOR ENABLE
0764 1272 ENBINT : ENABLE
0764 1273 JMP MTPR (SP)+,S^#PR$_IPL : OTHERWISE LET EXEC HANDLE
0764 1274 .IFF EXESTBIT : FALSE IF PROCESS VERSION
0764 1275 CLRL R0 : RESIGNAL
0764 1276 RET : UNEXPECTED TBIT EXCEPTION
0764 1277 .ENDC :
0764 1278 XDELDBG: : COMMON WITH DEBUG EXCEPTION
0764 1279 BICL #<1@PSL$V_TBIT>,SAVPSL-B(R11) : CLEAR TBIT IN PSL
0764 1280 BSBB UNBRK : REPLACE OPCODES
0764 1281 BBSC #V_ATBRK,(R10),PROCEED : CHECK FOR PROCEED
0764 1282 :
0764 1283 : OUTPUT STEP MESSAGE
0764 1284 :
0764 1285 :
0764 1285 MOVL SAVPC-B(R11),CURDOT-B(R11) : SET ADDRESS
0764 1285 IFNORD #4,@CURDOT-B(R11),GETCMD : SKIP DISPLAY IF NOT READABLE
0764 1286 PROBER #0,#4,@CURDOT-B(R11)
0764 1287 BEQL GETCMD
0764 1287 BSBW LOCPROMPT : PROMPT WITH ADDRESS/CONTENT
0764 1288 BRB GETCMD : GO GET COMMANDS
0764 1289

```

```

0795 1291 .SBTTL UNBRK - RESTORE OPCODES FOR BREAKPOINTS
0795 1292 :
0795 1293 : UNBRK
0795 1294 :
0795 1295 UNBRK:
50 51 08 D0 0795 1296 MOVL #NBRK,R1 : INIT LOOP
F91B CF41 D0 0798 1297 10$: MOVL BRKADR[R1],R0 : GET BREAKPOINT ADDRESS
06 13 079E 1298 BEQL 20$ : SKIP IF NOT ENABLED
07A0 1299 .IF DF,SW,PROCESS :
07A0 1300 PUSHR #^M<R0,R1,R2,R3,R4,R5> : SAVE REGS FOR PROTECTION CHANGE
07A0 1301 MOVL R0,R4 : FORM INADR RANGE FOR SET PROTECTION
07A0 1302 MOVL R0,R5 :
07A0 1303 BSBW SETWRT : SET PAGE WRITABLE
07A0 1304 MOVQ (SP),R0 : RESTORE R0,R1
60 F936 CF41 90 07A0 1305 .ENDC :
07A0 1306 MOVB BRKOP[R1],(R0) : RESTORE OPCODE
07A6 1307 .IF DF,SW,PROCESS :
07A6 1308 BSBW REPROT : RESTORE PROTECTION
07A6 1309 POPR #^M<R0,R1,R2,R3,R4,R5> : RESTORE REGISTERS
EF 51 F5 07A6 1310 .ENDC :
05 07A9 1311 20$: SOBGTR R1,10$ : DO THEM ALL
07AA 1312 RSB : AND RETURN
07AA 1313

```

```

07AA 1315 .SBTTL SETBRK - SET BREAK POINT INSTRUCTIONS
07AA 1316 :
07AA 1317 : SETBRK
07AA 1318 :
51 08 D0 07AA 1319 SETBRK: MOVL #NBRK,R1 : INIT LOOP
50 F906 CF41 D0 07AD 1320 10$: MOVL BRKADR[R1],R0 : GET ADDRESS
14 13 07B3 1321 : BEQL 20$ : SKIP IF NOT ENABLED
F920 CF41 60 90 07B5 1322 : MOVB (R0),BRKOP[R1] : SAVE OPCODE
6A 18 93 07BB 1323 : BITB #<<1@V_TBIT>!<1@V_ATBRK>>, (R10) : CHECK FOR TRACE
06 13 07BE 1324 : BEQL 15$ : NO TRACE, SET ANYWAY
54 AB 50 D1 07C0 1325 : CML R0,SAVPC-B(R11) : CHECK FOR AT BPT
03 13 07C4 1326 : BEQL 20$ : YES, DONT SET IT
07C6 1327 15$:
07C6 1328 .IF DF,SW PROCESS
07C6 1329 PUSH R #*M<R0,R1,R2,R3,R4,R5> : SAVE REGISTERS FOR PROTECTION CHANGE
07C6 1330 MOVL R0,R4 : SET START ADDRESS OF RANGE
07C6 1331 MOVL R0,R5 : AND END ADDRESS
07C6 1332 BSBW SETWRT : SET PAGE WRITABLE
07C6 1333 MOVL (SP),R0 : RESTORE BPT ADDRESS
60 03 90 07C6 1334 .ENDC
07C6 1335 MOVB #3,(R0) : SET BREAKPOINT OPCODE
07C9 1336 .IF DF,SW PROCESS
07C9 1337 BSBW REPROT : RESTORE ORIGINAL PROTECTION VALUE
07C9 1338 POP R #*M<R0,R1,R2,R3,R4,R5> : AND REGISTERS
07C9 1339 .ENDC
E1 51 F5 07C9 1340 20$: SOBGTR R1,10$ : DO THEM ALL
05 07CC 1341 : RSB : AND RETURN
07CD 1342

```

```

          07CD 1344      .SBTTL GETBPTX - GET INDEX FOR BREAKPOINT
          07CD 1345      :
          07CD 1346      : GETBPTX
          07CD 1347      :
          07CD 1348      GETBPTX:
F8E1 CF43 53 08 D0 07CD 1349      MOVL #NBRK,R3      : INIT LOOP
          54 AB D1 07D0 1350 10$:  CMPL SAVPC-B(R11),BRKADR[R3] : IS THIS A BPT?
          03 13 07D7 1351      BEQL 20$      : YES
          F4 53 F5 07D9 1352      SOBGTR R3,10$ : NO, CONTINUE
          05 07DC 1353 20$:      RSB      : RETURN

```

XDELTA  
SYN  
ADD  
AS  
B  
BLA  
BMS  
BR  
BR  
BR  
BR  
BS  
CL  
CL  
CL  
CL  
COL  
CO  
CO  
CO  
CR  
CUI  
CUI  
DCC  
DI  
DO  
DQ  
DT  
EN  
EN  
EQ  
EQ  
ER  
ER  
ER  
ES  
EX  
EX  
EX  
EX  
EX  
F1  
F3  
F4  
F5  
C  
C  
G  
GE  
GL  
GO

```

07DD 1355 .SBTTL QUOTE - INPUT CHARACTER STRING
07DD 1356 :
07DD 1357 :
07DD 1358 :
07DD 1359 QUOTE:
55 6B D0 07DD 1360 MOVL CURDOT-B(R11),R5 : POINT TO STRING BUFFER
FBDB 30 07E0 1361 5$: BSBW GETCHAR : GET CHARACTER
58 27 91 07E3 1362 CMPB #QUOT,R8 : CHECK FOR QUOTE
05 13 07E6 1363 BEQL 10$ : YES, END OF STRING
85 58 90 07E8 1364 MOVB R8,(R5)+ : INSERT IN BUFFER
F3 11 07EB 1365 BRB 5$ : AND CONTINUE
6B 55 D0 07ED 1366 10$: MOVL R5,CURDOT-B(R11) : SAVE NEW DOT
05 07F0 1367 RSB : RETURN

```

XDE  
Sym  
PR  
PR  
PR  
PR  
PR  
PR  
PR  
PR  
PR  
PR  
PR  
PR  
PR  
PR  
PR  
PR  
PR  
PR  
PR  
PR  
PSI  
PSI  
PSI  
QU  
QU  
QU  
QU  
RDE  
RDE  
REC  
REC  
REC  
REC  
REC  
REC  
RSE  
RUE  
SAI  
SAI  
SAI  
SAI  
SAI  
SAI  
SAI  
SAI  
SAI  
SAI  
SAI  
SCA  
SCI  
SCI  
SCI  
SEI  
SEI  
SEI  
SH  
SH  
SL  
SL  
ST  
ST  
SU  
TAI

```

07F1 1369 .SBTTL DEPOSIT
07F1 1370 :
07F1 1371 : DEPOSIT DATA
07F1 1372 :
07F1 1373 DEPOSIT:
1D 6A 1F E0 07F1 1374 BBS #V_PREG,(R10),40$ : BR IF PROCESSOR REGISTER
07F5 1375 .IF DF_SW_PROCESS :
07F5 1376 MOVL CURDOT-B(R11),R4 : GET CURRENT DOT
07F5 1377 TSTL PID-B(R11) : CHECK FOR ARBITRARY PROCESS DEPOSIT
07F5 1378 BNEQ 50$ : BR IF YES
07F5 1379 .ENDC
07F5 1380 CASE CURTYPE-B(R11),TYPE=B,<- : SWITCH ON TYPE
07F5 1381 10$,- : BYTE
07F5 1382 20$,- : WORD
07F5 1383 30$,- : LONG
07F5 1384 >
02' 00 FE AB 8F 07F5 30019$: CASEB CURTYPE-B(R11),#0,S^#<<30020$-30019$>/2>-1
07FA 30019$: .SIGNED_WORD 10$-30019$
0006' 07FA 30019$: .SIGNED_WORD 20$-30019$
000C' 07FC 30019$: .SIGNED_WORD 30$-30019$
0012' 07FE
0800 30020$:
0800 1385 .IF NDF_SW_PROCESS :
00 BB E0 AB 90 0800 1386 10$: MOVB F1-B(R11),@CURDOT-B(R11) : STORE BYTE
05 0805 1387 RSB : RETURN
00 BB E0 AB 80 0806 1388 20$: MOVW F1-B(R11),@CURDOT-B(R11) : STORE WORD
05 0808 1389 RSB : RETURN
00 BB E0 AB D0 080C 1390 30$: MOVL F1-B(R11),@CURDOT-B(R11) : STORE LONG
05 0811 1391 RSB : RETURN
68 E0 AB DA 0812 1392 40$: MTPR F1-B(R11),CURDOT-B(R11) : SET VALUE IN PROCESSOR REGISTER
05 0816 1393 RSB
0817 1394 .IFF : FALSE IF PROCESS VERSION
0817 1395 10$: : BYTE DEPOSIT
0817 1396 MOVL R4,R5 : START AND END ADDRESSES EQUAL
0817 1397 BSBW SETWRT : SET WRITABLE, OLD PROT TO R2
0817 1398 MOVB F1-B(R11),(R4) : STORE BYTE
0817 1399 BSBW REPROT : RESTORE PROTECTION
0817 1400 RSB
0817 1401
0817 1402 20$: ADDL3 #1,R4,R5 : WORD DEPOSIT, FORM END ADDRESS
0817 1403 BSBW SETWRT : SET WRITABLE
0817 1404 MOVW F1-B(R11),(R4) : STORE WORD
0817 1405 BSBW REPROT : RESTORE PROTECTION
0817 1406 RSB
0817 1407
0817 1408 30$: ADDL3 #3,R4,R5 : LONGWORD DEPOSIT, FORM END ADDRESS
0817 1409 BSBW SETWRT : SET WRITABLE
0817 1410 MOVL F1-B(R11),(R4) : STORE LONG WORD
0817 1411 BSBW REPROT : RESTORE PROTECTION
0817 1412 RSB
0817 1413
0817 1414 40$: $CMKRNLS B^DEPPREG,(AP) : PROCESSOR REGISTER
0817 1415 RSB : DEPOSIT IN PROCESSOR REGISTER
0817 1416
0817 1417 50$: : DEPOSIT IN ARBITRARY PROCESS
0817 1418 CASE CURTYPE-B(R11),TYPE=B,<- : SWITCH ON TYPE
0817 1419 60$,- : BYTE

```

```

0817 1420          70$,-          : WORD
0817 1421          80$>         : LONGWORD
0817 1422          :
0817 1423 60$:    RSB           :
0817 1424          PUSHAB W^DPBYTE : SET ADDRESS OF BYTE ROUTINE
0817 1425 70$:    BRB           90$ :
0817 1426          PUSHAB W^DPWORD : SET ADDRESS OF WORD ROUTINE
0817 1427 80$:    BRB           90$ :
0817 1428 90$:    PUSHAB W^DPLONG : SET ADDRESS OF LONG ROUTINE
0817 1429          PUSHL PID-B(R11) : SET PID OF TARGET
0817 1430          PUSHL CURDOT-B(R11) : ADDRESS FOR STORE
0817 1431          PUSHL F1-B(R11)  : VALUE TO STORE
0817 1432          PUSHL #4         : ARGUMENT COUNT
0817 1433          MOVL SP,R0       : POINTER TO ARGUMENT LIST
0817 1434          TSTL MFYFLG-B(R11) : CHECK FOR STORE ENABLED
0817 1435          BEQL 100$        : BR IF NOT
0817 1436 100$:   $CMKRNLS W^QGET,(R0) : CALL TO QUEUE REQUEST
0817 1437          ADDL #20,SP      : CLEAN STACK
0817 1438          RSB             : AND RETURN
0817 1439 DEPPREG: .WORD 0         : DEPOSIT INTO PROCESSOR REGISTER
0817 1440          MOVAB W^PREXC,(FP) : SET EXCEPTION HANDLER
0817 1441          MTPR F1-B(R11),CURDOT-B(R11) : PLACE FIELD VALUE IN REG
0817 1442          MOVL #1,R0       : RETURN SUCCESS
0817 1443          RET             :
0817 1444          :
0817 1445 PREXC:  .WORD 0         : PROCESSOR REGISTER EXCEPTION HANDLER
0817 1446          ADDL3 #4,8(AP),R1 : POINT TO EXCEPTION FP
0817 1447          MOVL (R1),12(FP)  : SET AS RETURN FP
0817 1448          MOVAB B^10$,16(FP) : SET RETURN ADDRESS
0817 1449 10$:   MOVZWL #1,R0      : SET NORMAL STATUS
0817 1450          RET             : AND RETURN
0817 1451          :
0817 1452          .ENDC

```

```
0817 1454 .SBTTL EXECUTE - PERFORM COMMAND STRING
0817 1455 :
0817 1456 : EXECUTE
0817 1457 :
0817 1458 EXECUTE:
09 6A 08 E1 0817 1459 BBC #V F1,(R10),10$ :
59 EO AB D0 081B 1460 MOVL F1=B(R11),R9 : EXIT IF NO ADDRESS
03 12 081F 1461 BNEQ 10$ : SET CHAR STRING
F981 31 0821 1462 BRW 10$ : NOT NULL
05 0824 1463 10$: RSB : SUPER RESET
0825 1464 : RETURN
```



```

00 6A  OF  E2  0825  1466  .SBTTL P - PROCESSOR REGISTER PREFIX
05      05  0825  1467  SET PROCESSOR REGISTER MODE
          05  0825  1468  :
          05  0825  1469  :
          05  0825  1470  PREG:
          05  0825  1471  BBSS #V_PMODE,(R10),10$ : PROCESSOR REGISTER MODE
          05  0829  1472  RSB      : SET PROCESSOR REG FLAG
          : RETURN

```

EXE  
Moc  
MS  
SYS

```

082A 1474      .SBTTL  PROCESS DEBUGGER INITIALIZATION
082A 1475
082A 1476      .IF    DF,SW_PROCESS
082A 1477 SALUTE: .ASCIZ  <CR><LF>/DELTA Version x2.1/<CR><LF> ;
082A 1478
082A 1479 TEST:
082A 1480 XDT$START: : START ADDRESS OF IMAGE ENTRY
082A 1481      .WORD  0 : GLOBAL START ADDRESS FOR CLI DEBUG
082A 1482 DELTA_START: : START ADDRESS FOR DEBUGGER ENTRY
082A 1483      $WAKE_S : NULL WAKE AND
082A 1484      $HIBER_S : HIBERNATE TO GET SYNCHRONIZED
082A 1485      MOVAB  TERMASK,TERMASKD+4 : RELOCATE TERMINATOR MASK DESCR
082A 1486      MOVAB  TTSTR,TTNAMD+4 : RELOCATE DESCRIPTOR
082A 1487      MOVAB  EXIHANDL,EXIHADR :
082A 1488      MOVAB  EXITCODE,EXCODA : RELOCATE EXIT HANDLER ARGS
082A 1489      CALLG  (AP),B^INITCALL : GENERATE CALL FRAME
082A 1490      RET :
082A 1491
082A 1492 NOBRK: MOVL  4(AP),AP : GET EXCEPTION ARGUMENT LIST
082A 1493      BRW   EXCEPT+2 : AND GOTO EXCEPTION HANDLER
082A 1494
082A 1495 INITCALL:
082A 1496      .WORD  0 : ENTRY MASK
082A 1497      MOVAB  W^CATCHALL,(FP) : SET CATCHALL EXCEPTION HANDLER
082A 1498      $DCLEXH_S EXITBLK : DECLARE USER MODE EXIT HANDLER
082A 1499      $CMKRNL_S W^SETEXC,(AP) : SET EXCEPTION VECTORS
082A 1500      $SETEXV_S ADDRES=W^EXCEPT, :
082A 1501      ACMODE=#3,- :
082A 1502      VECTOR=#0 : SET PRIMARY FOR USER
082A 1503      $SETEXV_S ADDRES=W^CATCHALL,- : SET LAST CHANCE HANDLER
082A 1504      ACMODE=#3,- : FOR USER MODE
082A 1505      VECTOR=#2 : SPECIFY LAST CHANCE HANDLER
082A 1506      $ASSIGN_S TTNAMD,TTCHAN : ASSIGN DEVICE
082A 1507      BLBS  -R0,10$ : CONTINUE IF SUCCESS
082A 1508      RET : ELSE EXIT WITH ERROR CODE IN R0
082A 1509 10$: MOVAB  SALUTE,R4 : SET ADDRESS OF SALUTATION
082A 1510      BSBW  OUTZSTRING : OUTPUT IT
082A 1511      BBS   #CLI$V_DBGEXCP,24(AP),NOBRK : BR IF LATER INVOCATION
082A 1512 : VIA $DEBUG COMMAND
082A 1513      CALLG  (AP),B^20$ : CREATE TOP CALL FRAME
082A 1514      RET :
082A 1515 20$: .WORD  0 : NULL ENTRY MASK
082A 1516      ADDL  #4,4(AP) : ADVANCE STARTING ADDRESS POINTER
082A 1517      MOVPSL -(SP) : SAVE PSL
082A 1518      ADDL3 #2,@4(AP),-(SP) : FETCH CURRENT STARTING ADDRESS
082A 1519      MOVZWL #SS$_DEBUG,-(SP) : SET EXCEPTION CODE
082A 1520      PUSHL #3 : SIGNAL ARG COUNT
082A 1521      MOVL  SP,R0 : SAVE POINTER
082A 1522      MOVQ  R0,-(SP) : SAVE PHONY R0,R1
082A 1523      PUSHL #0 : DEPTH
082A 1524      PUSHL FP : FP
082A 1525      PUSHL #4 : ARG COUNT
082A 1526      PUSHL SP : POINTER TO MECH
082A 1527      PUSHL R0 : POINTER TO SIGNAL
082A 1528      CALLS #2,W^EXCEPT : SIGNAL PHONY EXCEPTION
082A 1529      ADDL  #12,SP : CLEAN BACK TO R0,R1
082A 1530      MOVQ  (SP)+,R0 : RESTORE R0,R1

```

```

082A 1531      ADDL  #8,SP      ; CLEAN BACK TO PC,PSL
082A 1532      REI          ; RETURN TO TARGET PROGRAM
082A 1533
082A 1534
082A 1535      SETEXC: .WORD 0      ; ENTRY MASK
082A 1536      $SETEXV_S      ADDRESS=B^EXCEPT,-
082A 1537      PRVHND=KCOND,-
082A 1538      ACMODE=#0      ; SET KERNEL
082A 1539      $SETEXV_S      ADDRESS=W^CATCHALL,-
082A 1540      ACMODE=#0,-    ; SET KERNEL MODE LAST CHANCE HANDLER
082A 1541      VECTOR=#2     ; SPECIFY LAST CHANCE VECTOR
082A 1542      -----
082A 1543      $SETEXV_S      ADDRESS=B^EXCEPT,-
082A 1544      PRVHND=ECOND,-
082A 1545      ACMODE=#1     ; SET EXEC MODE EXCEPTION HANDLER
082A 1546      $SETEXV_S      ADDRESS=W^CATCHALL,-
082A 1547      ACMODE=#1,-   ; SET EXEC MODE LAST CHANCE HANDLER
082A 1548      VECTOR=#2     ; SPECIFY LAST CHANCE VECTOR
082A 1549      -----
082A 1550      $SETEXV_S      ADDRESS=B^EXCEPT,-
082A 1551      PRVHND=SCOND,-
082A 1552      ACMODE=#2     ; SET SUPERVISOR MODE EXCEPTION HANDLER
082A 1553      $SETEXV_S      ADDRESS=W^CATCHALL,-
082A 1554      ACMODE=#2,-   ; SET SUPERVISOR LAST CHANCE HANDLER
082A 1555      VECTOR=#2     ; SPECIFY LAST CHANCE VECTOR
082A 1556      RET
082A 1557
082A 1558      EXCEPT: .WORD 0 ; EXCEPTION HANDLER ENTRY MASK
082A 1559      $SETEXV_S      ADDRESS=B^EXCEPT,-
082A 1560      ACMODE=#3,-
082A 1561      VECTOR=#0     ; RE-ESTABLISH USER PRIMARY VECTOR
082A 1562      ADDL3 #4,4(AP),R0 ; GET POINTER TO SIGNAL
082A 1563      MOVPSL R1      ; GET CURRENT PSL
082A 1564      EXTZV #PSL$V_CURMOD,#PSL$$_CURMOD,R1,R1 ;
082A 1565      BBSS R1,DBGACTIVE,40$ ; BR IF ALREADY ACTIVE
082A 1566      CMPL #SS$_TBIT,(R0) ; IS IT TBIT?
082A 1567      BNEQ 10$      ; NO
082A 1568      5$: BRW XDELTBIT ; YES, A TBIT
082A 1569      10$: CMPL #SS$_BREAK,(R0) ; IS IT BREAKPOINT?
082A 1570      BNEQ 20$      ; NO
082A 1571      15$: BRW XDELBPT ; YES, A BREAKPOINT
082A 1572      20$:          ; SOME OTHER EXCEPTION
082A 1573      CMPL #SS$_UNWINDING,(R0) ; IS IT UNWINDING
082A 1574      BEQL 60$      ; YES
082A 1575      CMPL #SS$_COMPAT,(R0)+ ; IS IT COMPATIBILITY MODE EXCEPT?
082A 1576      BNEQ 30$      ; NO
082A 1577      CMPL #1,(R0) ; IS IT COMPATIBILITY BPT?
082A 1578      BEQL 15$      ; YES
082A 1579      CMPL #7,(R0) ; IS IT COMPATIBILITY TBIT?
082A 1580      BEQL 5$       ; YES
082A 1581      30$: CMPL #SS$_DEBUG,-(R0) ; IS IT DEBUG EXCEPTION?
082A 1582      BNEQ 40$      ; NO
082A 1583      BSBW SAVE ; SAVE EVERYTHING
082A 1584      BRW XDELDDBG ; AND TREAT AS FUNNY BPT
082A 1585      40$:          ; UNEXPECTED EXCEPTION
082A 1586      BBCC R1,DBGACTIVE,50$ ; CLEAR DEBUG ACTIVE
082A 1587      50$: CLRL RO ; RETURN FALSE FOR RESIGNAL

```

```

082A 1588      RET
082A 1589 60$:  MOVL  #1,R0      ; IGNORE AND RESIGNAL
082A 1590      RET
082A 1591      .PAGE
082A 1592      .SBTTL  HANDLER FOR DEBUG EXCEPTIONS
082A 1593
082A 1594 DBGEXCEP:
082A 1595      .WORD  0
082A 1596      ADDL3 #4,8(AP),R1  ; POINT TO EXCEPTION FP
082A 1597      MOVL  FP,R0      ; INIT LINK FOR CALL FRAMES
082A 1598 10$:  CMPL  12(R0),(R1) ; IS THIS THE LAST ONE?
082A 1599      BEQL  20$
082A 1600      MOVAB B^30$,16(R0) ; SET FOR RETURN
082A 1601      MOVL  12(R0),R0
082A 1602      BRB   10$
082A 1603 20$:  MOVAB  XDELACV,16(R0) ; SET RETURN FOR ERROR
082A 1604 30$:  RET
082A 1605
082A 1606 CATCHALL:
082A 1607      .WORD  0
082A 1608      MOVPSL R1
082A 1609      EXTZV #PSL$V_CURMOD,#PSL$$_CURMOD,R1,R1 ; ISOLATE CURRENT MODE
082A 1610      BBSC  R1,DBGACTIVE,10$ ; MUST NOT BE DEBUGGER EXCEPTION
082A 1611      CLRL  R0
082A 1612      RET
082A 1613 10$:  BSBW  SAVE
082A 1614      ADDL3 #4,4(AP),R0 ; SAVE EVERYTHING
082A 1615      MOVL  (R0),R3 ; POINT TO EXCEPTION CODE
082A 1616      BSBW  CRLF ; GET IT
082A 1617      BSBW  OUTLONG ; OUTPUT CR/LF
082A 1618      MOVAB B^EXCMMSG,R4 ; OUTPUT EXCEPTION CODE
082A 1619      BSBW  OUTZSTRING ; OUTPUT MESSAGE
082A 1620      BRW   XDELDBG ; TEXT FOR EXCEPTION
082A 1621 EXCMMSG: .ASCIZ / EXCEPTION / ; AND DISPLAY INSTRUCTION
082A 1622
082A 1623 EXIHANDLE:
082A 1624      .WORD  0
082A 1625      BITB  #15,DBGACTIVE ; EXIT HANDLER
082A 1626      BEQL  10$ ; ENTRY MASK
082A 1627      $CMKRNLS CLREXV,(AP) ; TEST FOR DEBUG ACTIVE IN ANY MODE
082A 1628      MOVL  @4(AP),R0 ; NO, REPORT EXIT
082A 1629      RET ; RESET EXCEPTION VECTORS
082A 1630 10$:  ; RESTORE
082A 1631      MOVPSL -(SP) ; RETURN
082A 1632      PUSHL 16(FP) ; PROGRAM EXIT
082A 1633      PUSHL @4(AP) ; BUILD EXCEPTION FRAME
082A 1634      PUSHL #3 ; EXIT CODE FOR EXCEPTION CODE
082A 1635      PUSHR #^M<R0,R1> ; ARG COUNT
082A 1636      MOVQ  AP,-(SP)
082A 1637      PUSHL #4 ; MECHANISM COUNT
082A 1638      PUSHL SP ; POINTER TO MECHANISM
082A 1639      PUSHAL 24(SP) ; POINTER TO SIGNAL
082A 1640      PUSHL #2
082A 1641      MOVL  SP,AP ; SET AP FOR EXCEPTION
082A 1642      BSBW  SAVE ; SAVE EVERYTHING
082A 1643      MOVAB B^EXIMSG,R4 ; DISPLAY EXIT MESSAGE
082A 1644      BSBW  OUTZSTRING ; OUTPUT TEXT

```

Sym  
---  
BUC  
CLU  
EXE  
EXE  
EXE  
EXE  
EXE  
IOC  
MMC  
PMS  
PMS  
PMS  
SCS  
SCS  
SCS  
SCS  
SCS  
SCS  
SCS  
SCS  
SYS

```

082A 1645      MOVL   SAVAP-B(R11),R3      : GET POINTER TO EXCEPTION ARGLIST
082A 1646      MOVL   4(R3),R3      : GET EXIT CODE ADDRESS
082A 1647      BSBW   OUTLONG      : DISPLAY IT
082A 1648      $DCLEXH_S   EXITBLK   : RE-ESTABLISH EXIT HANDLER
082A 1649      MOVPSL_R1      : GET CURRENT PSL
082A 1650      EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,R1,R1 : GET CURRENT MODE
082A 1651      BBSS   R1,DBG$ACTIVE,20$ : SET DELTA ACTIVE FOR MODE
082A 1652 20$:   BRW     XDÉLDBG      :
082A 1653
082A 1654 EXIMSG: .ASCIZ <CR><LF>/ EXIT / :
082A 1655
082A 1656 CLREXV:      : CLR EXCEPTION VECTORS
082A 1657      .WORD   0              : ENTRY MASK
082A 1658      $SETEXV_S   ADDRESS=@KCOND,- :
082A 1659      ACMODE=#0      :
082A 1660      $SETEXV_S   ADDRESS=@ECOND,- :
082A 1661      ACMODE=#1      :
082A 1662      $SETEXV_S   ADDRESS=@SCOND,- :
082A 1663      ACMODE=#2      :
082A 1664      RET              :
082A 1665
082A 1666      .PAGE
082A 1667      .SBTTL  SETWRT - SET PAGES WRITABLE
082A 1668
082A 1669 SETWRT:      :
082A 1670      MOVAL   -(SP),R2      : ADDRESS FOR RETURN OF PROT
082A 1671      $CMKRNL_S   B^SETPRTK,(R2) :
082A 1672      BLBS   R0,10$      : CONTINUE IF NO ERROR
082A 1673      CALLG   (R2),B^SETPRTK :
082A 1674 10$:   POPR    #^M<R2>    : RESTORE PROTECTION VALUE
082A 1675      RSB      : RETURN
082A 1676
082A 1677 SETPRTK: .WORD   0              :
082A 1678      MOVQ    R5,-(SP)      : INADR, START AND END ADDRESSES
082A 1679      MOVL   SP,R1        : ADDRESS OF INADR
082A 1680      $SETPRT_S   INADR=(R1),- :
082A 1681      PROT=#PRT$C_UW,- : WRITABLE BY ALL
082A 1682      ACMODE=#0,- :
082A 1683      PRVPRT=(AP)      : ADDRESS AT WHICH TO RETURN PROT
082A 1684      MOVL   #1,R0        : ALWAYS SUCCESS
082A 1685      RET              :
082A 1686
082A 1687 REPROT:      : RESTORE PROTECTION
082A 1688      RSB      :
082A 1689      .PAGE
082A 1690      .SBTTL  FETCHP - FETCH DATA FROM ANOTHER PROCESS
082A 1691 FETCHP: CASE CURTYPE-B(R11),TYPE=B,<- :
082A 1692      10$,- : 0 => BYTE
082A 1693      20$,- : 1 => WORD
082A 1694      30$> : 2 => LONG
082A 1695      RSB      : UNKNOWN
082A 1696 10$:   PUSHAB W^FPBYTE    : SET FOR BYTE FETCH
082A 1697      BRB     40$          :
082A 1698 20$:   PUSHAB W^FPWORD    : SET FOR WORD FETCH
082A 1699      BRB     40$          :
082A 1700 30$:   PUSHAB W^FPLONG    : SET FOR LONGWORD FETCH
082A 1701 40$:   PUSHL   PID-B(R11) : P. OF TARGET PROCESS

```

```

082A 1702 PUSHAB QUAN-B(R11) ; SET ADDRESS TO RETURN VALUE
082A 1703 PUSHL CURDOT-B(R11) ; AND ADDRESS OF VALUE
082A 1704 PUSHL #4 ; ARGUMENT COUNT
082A 1705 MOVL SP,R0 ; SAVE POINTER TO ARG LIST
082A 1706 $CMKRNLS W^QGET,(R0) ; Q AST FOR DATA FETCH
082A 1707 BLBC R0,50$ ; BR IF FAILED
082A 1708 $HIBERS ; WAIT FOR DATA TO RETURN
082A 1709 50$: ADDL #20,SP ; CLEAN STACK
082A 1710 RSB ; AND RETURN DATA
082A 1711 .PAGE
082A 1712 .SBTTL QGET - QUEUE AST TO GET DATA FROM ANOTHER PROCESS
082A 1713 :
082A 1714 : INPUTS: 04(AP) - LOCATION OF DATA
082A 1715 : 08(AP) - RETURN LOCATION
082A 1716 : 12(AP) - PID OF TARGET PROCESS
082A 1717 : 16(AP) - CODE SEGMENT POINTER
082A 1718 :
082A 1719 FP_ORIGPID=ACBSL_AST
082A 1720 FP_ADDR=ACBSL_ASTPRM
082A 1721 FP_VALUE=ACBSL_ASTPRM
082A 1722 FP_RETLOC=ACBSL_KAST+4
082A 1723 QGET: .WORD ^M<R2,R3,R4,R5> ; ENTRY MASK
082A 1724 MOVZWL #SS$ NONEXPR,R0 ; ASSUME BAD PIX
082A 1725 CMPW 12(AP),@#SCH$GL_MAXPIX ; CHECK PIX FOR LEGAL PROCESS
082A 1726 BGTR 10$ ; BR IF NOT
082A 1727 MOVZWL @16(AP),R1 ; GET SIZE OF CODE SEGMENT
082A 1728 MOVAB IRP$C_LENGTH(R1),R1 ; ADD SIZE OF PACKET DATA
082A 1729 JSB @#EXES$ALLOCBUF ; ALLOCATE BUFFER TO CONTAIN CODE
082A 1730 BLBC R0,10$ ; BRANCH IF NONE
082A 1731 MOVL R2,R5 ; SAVE ADDRESS OF PACKET
082A 1732 PCBSL_PID(R4),FP_ORIGPID(R5) ; SET PID FOR RETURN
082A 1733 MOVB #^X80,ACBSB_RMOD(R5) ; SET FOR SPECIAL KERNEL AST
082A 1734 MOVAB ACBSL_KAST+8(R5),ACBSL_KAST(R5) ; SET ADDRESS FOR AST
082A 1735 MOVL 4(AP),FP_ADDR(R5) ; SET ADDRESS FOR FETCH
082A 1736 MOVL 8(AP),FP_RETLOC(R5) ; AND ADDRESS OF RETURN LOCATION
082A 1737 MOVL 16(AP),R0 ; GET ADDRESS OF CODE SEGMENT
082A 1738 MOVL 12(AP),ACBSL_PID(R5) ; SET TARGET PID
082A 1739 PUSHR #^M<R0,R1,R2,R3,R4,R5> ; SAVE REGS FOR MOVC
082A 1740 MOVC3 (R0)+,(R0),ACBSL_KAST+8(R5) ; COPY CODE SEGMENT TO BUFFER
082A 1741 POPR #^M<R0,R1,R2,R3,R4,R5> ; RESORE REGISTERS
082A 1742 MOVZBL #PRIS_TICOM,R2 ; SET PRIORITY INCREMENT CLASS
082A 1743 JSB @#SCH$QAST ; QUEUE AST FOR TARGET
082A 1744 10$: RET ; RETURN TO ORIGINAL MODE
082A 1745
082A 1746 .SBTTL FPBYTE - FETCH BYTE FROM PROCESS
082A 1747 FPBYTE: .WORD 90$--2 ; SIZE OF CODE SEGMENT
082A 1748 IFNORD #1,@FP_ADDR(R5),10$ ; BRANCH IF NOT READABLE
082A 1749 MOVB @FP_ADDR(R5),FP_VALUE(R5) ; GET VALUE
082A 1750 10$: MOVL FP_ORIGPID(R5),ACBSL_PID(R5) ; SET PID FOR RETURN AST
082A 1751 MOVB #^X80,ACBSB_RMOD(R5) ; SET FOR KAST AGAIN
082A 1752 MOVAB B^20$,ACBSL_KAST(R5) ; SET NEW AST ADDRESS
082A 1753 MOVZBL #PRIS_TICOM,R2 ; SET PRIORITY INCREMENT CLASS
082A 1754 JMP @#SCH$QAST ; QUEUE RETURN AST
082A 1755 20$: IFNOWRT #1,@FP_RETLOC(R5),30$ ; IF NOT WRITABLE THEN SKIP IT
082A 1756 MOVB FP_VALUE(R5),@FP_RETLOC(R5) ; RETURN VALUE
082A 1757 30$: MOVL ACBSL_PID(R5),R1 ; GET PID FOR WAKE
082A 1758 SETIPL #IPL$_SYNCH ; RAISE TO SYNCH

```

Vir  
Sta  
Ima  
Ima  
Ima  
Nur  
Nur  
Nur  
Nur  
Nur  
Ima  
Max  
Est

Per  
---

Tot  
Us1  
Tot  
Nur  
23  
A 1  
LIM  
CT  
BAS

```

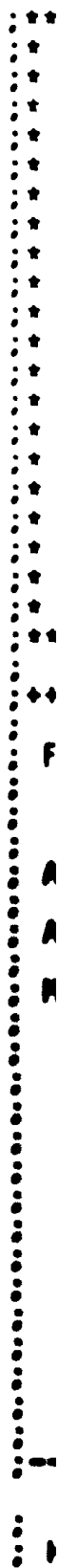
082A 1759 JSB @#SCH$WAKE ; WAKE PROCESS
082A 1760 SETIPL #IPL$ASTDEL ; LOWER IPL
082A 1761 MOVL R5,R0 ; SET ADDRESS FOR RELEASE
082A 1762 JMP @#EXE$DEANONPAGED ; FREE BLOCK AND EXIT
082A 1763 90$: ; END OF CODE SEGMENT
082A 1764
082A 1765 .PAGE
082A 1766 .SBTTL DPBYTE - DEPOSIT BYTE TO PROCESS
082A 1767 DPBYTE: .WORD 90$--2 ; SIZE OF CODE SEGMENT
082A 1768 20$: IFNOWRT #1,@FP_RETLOC(R5),30$ ; IF NOT WRITABLE THEN SKIP IT
082A 1769 MOVB FP_VALUE(R5),@FP_RETLOC(R5) ; RETURN VALUE
082A 1770 30$: MOVL R5,R0 ; SET ADDRESS FOR RELEASE
082A 1771 JMP @#EXE$DEANONPAGED ; FREE BLOCK AND EXIT
082A 1772 90$: ; END OF CODE SEGMENT
082A 1773
082A 1774 .PAGE
082A 1775 .SBTTL FPWORD - FETCH WORD FROM PROCESS
082A 1776 FPWORD: .WORD 90$--2 ; SIZE OF CODE SEGMENT
082A 1777 IFNORD #2,@FP_ADDR(R5),10$ ; BRANCH IF NOT READABLE
082A 1778 MOVW @FP_ADDR(R5),FP_VALUE(R5) ; GET VALUE
082A 1779 10$: MOVL FP_ORIGPID(R5),ACB$$_PID(R5) ; SET PID FOR RETURN AST
082A 1780 MOVAB #^X80,ACB$$_RMOD(R5) ; SET FOR KAST AGAIN
082A 1781 MOVAB B^20$,ACB$$_KAST(R5) ; SET FOR NEW AST ADDRESS
082A 1782 MOVZBL #PRI$TICOM,R2 ; SET PRIORITY INCREMENT CLASS
082A 1783 JMP @#SCH$QAST ; QUEUE RETURN AST
082A 1784 20$: IFNOWRT #2,@FP_RETLOC(R5),30$ ; IF NOT WRITABLE THEN SKIP IT
082A 1785 MOVW FP_VALUE(R5),@FP_RETLOC(R5) ; RETURN VALUE
082A 1786 30$: MOVL ACP$$_PID(R5),R1 ; GET PID FOR WAKE
082A 1787 SETIPL #IPL$SYNCH ; RAISE TO SYNCH
082A 1788 JSB @#SCH$WAKE ; WAKE PROCESS
082A 1789 SETIPL #IPL$ASTDEL ; LOWER IPL
082A 1790 MOVL R5,R0 ; SET ADDRESS FOR RELEASE
082A 1791 JMP @#EXE$DEANONPAGED ; FREE BLOCK AND EXIT
082A 1792 90$: ; END OF CODE SEGMENT
082A 1793
082A 1794 .PAGE
082A 1795 .SBTTL DPWORD - DEPOSIT WORD TO PROCESS
082A 1796 DPWORD: .WORD 90$--2 ; SIZE OF CODE SEGMENT
082A 1797 20$: IFNOWRT #2,@FP_RETLOC(R5),30$ ; IF NOT WRITABLE THEN SKIP IT
082A 1798 MOVW FP_VALUE(R5),@FP_RETLOC(R5) ; RETURN VALUE
082A 1799 30$: MOVL R5,R0 ; SET ADDRESS FOR RELEASE
082A 1800 JMP @#EXE$DEANONPAGED ; FREE BLOCK AND EXIT
082A 1801 90$: ; END OF CODE SEGMENT
082A 1802
082A 1803 .PAGE
082A 1804 .SBTTL FPLONG - FETCH LONG FROM PROCESS
082A 1805 FPLONG: .WORD 90$--2 ; SIZE OF CODE SEGMENT
082A 1806 IFNORD #4,@FP_ADDR(R5),10$ ; BRANCH IF NOT READABLE
082A 1807 MOVL @FP_ADDR(R5),FP_VALUE(R5) ; GET VALUE
082A 1808 10$: MOVL FP_ORIGPID(R5),ACB$$_PID(R5) ; SET PID FOR RETURN AST
082A 1809 MOVAB #^X80,ACB$$_RMOD(R5) ; SET FOR KAST AGAIN
082A 1810 MOVAB B^20$,ACB$$_KAST(R5) ; SET NEW KAST ADDRESS
082A 1811 CLRL R2 ; NULL PRIO INCR
082A 1812 JMP @#SCH$QAST ; QUEUE RETURN AST
082A 1813 20$: IFNOWRT #4,@FP_RETLOC(R5),30$ ; IF NOT WRITABLE THEN SKIP IT
082A 1814 MOVL FP_VALUE(R5),@FP_RETLOC(R5) ; RETURN VALUE
082A 1815 30$: MOVL ACP$$_PID(R5),R1 ; GET PID FOR WAKE

```

```

082A 1816      SETIPL #IPL$ SYNCH          ; RAISE TO SYNCH
082A 1817      JSB   @#SCH$WAKE        ; WAKE PROCESS
082A 1818      SETIPL #IPL$ _ASTDEL     ; LOWER IPL
082A 1819      MOVL  R5,RO              ; SET ADDRESS FOR RELEASE
082A 1820      JMP   @#EXE$DEANONPAGED ; FREE BLOCK AND EXIT
082A 1821 90$:                                     ; END OF CODE SEGMENT
082A 1822
082A 1823      .PAGE
082A 1824      .SBTTL DPLONG - DEPOSIT LONGWORD TO PROCESS
082A 1825 DPLONG: .WORD 90$--2          ; SIZE OF CODE SEGMENT
082A 1826 20$: IFNOWRT #4,@FP_RETLOC(R5),30$ ; IF NOT WRITABLE THEN SKIP IT
082A 1827      MOVL FP_VALUE(R5),@FP_RETLOC(R5) ; RETURN VALUE
082A 1828 30$: MOVL R5,RO              ; SET ADDRESS FOR RELEASE
082A 1829      JMP   @#EXE$DEANONPAGED ; FREE BLOCK AND EXIT
082A 1830 90$:                                     ; END OF CODE SEGMENT
082A 1831 DELEND:                                 ;
082A 1832      .ENDC
082A 1          :
082A 2          : NORMAL END STATEMENT WITHOUT START ADDRESS
082A 3          : USED TO ASSEMBLE XDELTA FOR EXEC DEBUGGING.
082A 4          :
082A 5          : .END

```





XDELTA  
Symbol table

- EXECUTIVE DEBUGGER

C 10

ADD	0000024A	R	02	HIGH	00000210	R	02
ASTEN	000000B8	R	02	INBUF	00000C04	R	02
B	00000058	R	02	INFLD	00000206	R	02
BLANK	00000431	R	02	INISBRK	*****	X	02
BMSG	000006EA	R	02	LBRACKET	000004B2	R	02
BRKADR	= 000000B8	R	02	LF	= 0000000A		
BRKCOM	= 00000100	R	02	LINEFEED	000002F2	R	02
BRKDSP	= 000000E0	R	02	LOCOUT	000002F9	R	02
BRKOP	= 000000DB	R	02	LOCP	00000465	R	02
BRKPOINT	000004D7	R	02	LOCPROMPT	000002F7	R	02
BSLSH	= 0000005C			MCHK	000005F0	R	02
CLR_730	00000607	R	02	MCHKSAV	00000164	R	02
CLR_750	00000607	R	02	MFYFLG	0000004C	R	02
CLR_780	00000602	R	02	MFYFLGS	00000589	R	02
CLR-END	0000060A	R	02	MMG\$PAGEFAULT	*****	X	02
COLON	0000057F	R	02	MODES	000004AE	R	02
COMMA	0000029B	R	02	MUL	00000242	R	02
CONTEXT	00000000	R	02	NBRK	= 00000008		
CONTEXTSZ	= 000000BC			NEGATE	0000043A	R	02
CR	= 0000000D			NEXTDOT	000002DF	R	02
CRLF	000003B4	R	02	NEXTLOC	000002F5	R	02
CURDOT	00000058	R	02	NEXTP	000001B5	R	02
CURTYPE	00000056	R	02	NMODES	= 00000004		
DCOM	0000019A	R	02	NPRIM	= 0000002A		
DEPOSIT	000007F1	R	02	NSEC	= 00000007		
DIV	00000246	R	02	NTERM	= 00000008		
DOT	0000058F	R	02	OPEN	00000256	R	02
DQUOTE	0000024E	R	02	OPER	00000057	R	02
DTYPE	00000055	R	02	OPERATOR	00000431	R	02
ENDEXPR	0000021F	R	02	OPERBAS	= 00000012		
ENDFIELD	0000029E	R	02	OUTB	= 00000006		
EQL1	0000046F	R	02	OUTBB	000002EF	R	02
EQUALS	00000468	R	02	OUTBSLSH	00000388	R	02
ERR2	000004AB	R	02	OUTBUF	00000060	R	02
ERR3	0000060A	R	02	OUTCHAR	00000391	R	02
ERR4	00000281	R	02	OUTCOM	00000364	R	02
ERROR	0000019E	R	02	OUTCR	= 00000004		
ESCAP	00000456	R	02	OUTDIGIT	0000035D	R	02
EXE\$ACVIOLAT	*****	X	02	OUTER	00000192	R	02
EXE\$BREAK	*****	X	02	OUTLONG	00000361	R	02
EXE\$GB_CPUTYPE	*****	X	02	OUTPUT	000002FE	R	02
EXE\$ROPRAND	*****	X	02	OUTPUTA	00000324	R	02
EXE\$TBIT	*****	X	02	OUTR8	0000038E	R	02
EXECUTE	00000817	R	02	OUTSPACE	000003AF	R	02
F1	00000038	R	02	OUTZBUF	0000037A	R	02
F2	0000003C	R	02	OUTZSTRING	0000037E	R	02
F3	00000040	R	02	PFNSAB_STATE	*****	X	02
F4	00000044	R	02	PFNSAB_TYPE	*****	X	02
F5	00000048	R	02	PFNSAL_BAK	*****	X	02
FCTR	00000054	R	02	PFNSAL_PTE	*****	X	02
FETCH	000002B9	R	02	PFNSAW_REFCNT	*****	X	02
CTBPTX	000007CD	R	02	PFNSAW_SWPVBN	*****	X	02
G_TCHAR	000003BE	R	02	PFNSAX_BLINK	*****	X	02
GETCMD	0000074C	R	02	PFNSAX_FLINK	*****	X	02
GETSCB	000006D8	R	02	PID	00000050	R	02
GLOBL	0000020A	R	02	PR\$_IPL	= 00000012		
GO	00000575	R	02	PR\$_KSP	= 00000000		

XDELTA  
Symbol table

- EXECUTIVE DEBUGGER

D 10

PR\$MAPEN	=	00000038		
PR\$MCE\$R	=	00000026		
PR\$RXCS	=	00000020		
PR\$RXDB	=	00000021		
PR\$SBIFS	=	00000030		
PR\$SCBB	=	00000011		
PR\$SID_TYP780	=	00000001		
PR\$TXCS	=	00000022		
PR\$TXDB	=	00000023		
PRET		00000217	R	02
PREG		00000825	RR	02
PRIMARY		00000168	RR	02
PROCD		0000057E	RR	02
PROCEED		00000751	RR	02
PROGCTR		000005A2	R	02
PSL\$S_CURMOD	=	00000002		
PSL\$V_CURMOD	=	00000018		
PSL\$V_TBIT	=	00000004		
QUAN		0000005C	R	02
QUANT		0000059C	R	02
QUOT	=	00000027		
QUOTE		000007DD	R	02
RDBUF	=	00000002		
RDCR	=	00000000		
REGCOM		000005BC	R	02
REGISTER		000005B4	R	02
RESET		00000477	R	02
RESTORE		00000680	R	02
RESTORR		00000685	R	02
RETURN		00000284	R	02
RSET		00000291	R	02
RUBOUT	=	0000007F		
SAVAP		000000A0	R	02
SAVE		0000060D	RR	02
SAVOCR		000000B4	RR	02
SAVPC		000000AC	RR	02
SAVPSL		000000B0	R	02
SAVR2		00000078	R	02
SAVRCR		000000B6	R	02
SAVREG		00000070	R	02
SAVRXCS		000000B8	R	02
SAVSP		000000A8	R	02
SCANP		000001B1	R	02
SCB\$AL_BASE	*****		X	02
SCH\$GL_CURPCB	*****		X	02
SCH\$GL_PCBVEC	*****		X	02
SECOND		00000484	R	02
SEMI		0000048B	RR	02
SETBRK		000007AA	R	02
SHFT		0000023D	RR	02
SHOBRK		00000533	R	02
SLASH		00000253	R	02
SLSH	=	0000002F		
STATUS		00000034	R	02
STEP		000004CD	RR	02
SUPERST		000001A5	R	02
TAB		00000446	R	02

TERM		00000181	R	02
UNBRK		00000795	RR	02
VALI		000005B1	RR	02
VALR		000005AE	RR	02
VALUE		000005A6	R	02
V_ASCII	=	00000001		
V_ATBRK	=	00000004		
V_F1	=	00000008		
V_F2	=	00000009		
V_F3	=	0000000A		
V_F4	=	0000000B		
V_F5	=	0000000C		
V_INFIELD	=	00000002		
V_NEGATE	=	00000007		
V_OPEN	=	00000000		
V_PREG	=	0000001F		
V_PRMODE	=	0000000F		
V_RUB	=	00000006		
V_TBIT	=	00000003		
V_TBITOK	=	00000005		
XDELACV		000005F0	R	02
XDELBPT		000006F4	RG	02
XDELDDBG		0000077B	R	02
XDELIBRK		000000BC	RG	02
XDELTBIT		00000764	RG	02
XDEL_LOADBASE		00000130	RG	02
XD\$GL_XESTRING		0000015C	RG	02
XD\$GL_XFSTRING		00000160	RG	02
XD\$GT_WORD_PFN	*****		X	02
XREG		000005E2	R	02
XREGV		00000124	R	02
XSET		000005D0	R	02

MSC  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$E  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$D  
\$E  
\$E  
\$E  
\$E  
\$E  
\$E  
\$E  
\$E  
\$E  
\$E  
\$E  
\$E  
\$E

...  
C

! 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
Z\$DEBUGXDELTA	0000082A ( 2090.)	02 ( 2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

! Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
Initialization	38	00:00:00.11	00:00:00.65
Command processing	124	00:00:01.02	00:00:04.74
Pass 1	405	00:00:15.16	00:00:44.35
Symbol table sort	0	00:00:02.05	00:00:03.57
Pass 2	338	00:00:05.33	00:00:12.81
Symbol table output	24	00:00:00.23	00:00:00.68
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	933	00:00:23.92	00:01:06.82

The working set limit was 1950 pages.  
87249 bytes (171 pages) of virtual memory were used to buffer the intermediate code.  
There were 70 pages of symbol table space allocated to hold 1237 non-local and 91 local symbols.  
1842 source lines were read in Pass 1, producing 18 object records in Pass 2.  
24 pages of virtual memory were used to define 23 macros.

! Macro library statistics !

Macro library name	Macros defined
_\$255\$DUA28:[MP.OBJ]MP.MLB;1	10
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	10
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	8
TOTALS (all libraries)	28

1396 GETS were required to define 28 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:XDELTA/OBJ=OBJ\$:XDELTA MSRCS:MPPREFIX/UPDATE=(ENHS:MPPREFIX)+MSRCS:XDELTA/UPDATE=(ENHS:XDELTA)+MSRCS:END/UPDATE=(ENHS

The grid contains 140 thumbnails. Several thumbnails are highlighted with larger text labels:

- MSCP
- MSCP MAP
- ADDUNIT LIS.
- MSCP LIS
- MPWATT LIS
- MPTIMER LIS
- XDELTA LIS
- MSCPDEF MAR