```
SOURCE ACCESS NAME=      PPC2.P359.SRC.CRUNCH
OBJECT ACCESS NAME=      PPC2.P359.OBJ.CRUNCHS
LISTING ACCESS NAME=     PPC2.P359.LST.CRUNCHS
ERROR ACCESS NAME=       .XMAERR
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=


LINE      KEY   NAME

 0002      A    VERSION
                    =>PPC2.P359.SRC.P359
```

```
0031                    IDT  'CRUNCH'
0032            ********************************************************************
0033            *
0034            *     CCCC     RRRRR    U    U   N      N    CCCC     H     H    *
0035            *    C    C    R    R   U    U   NN     N   C    C    H     H    *
0036            *    C         R    R   U    U   N N    N   C         H     H    *
0037            *    C         RRRRR    U    U   N  N   N   C         HHHHHH    *
0038            *    C         R  R     U    U   N   N  N   C         H     H    *
0039            *    C    C    R   R    U    U   N    NN   C    C    H     H    *
0040            *     CCCC     R    R    UUUU    N     N    CCCC     H     H    *
0041            *                                                                *
0042            *           PPPPP      3333     5555     9999                   *
0043            *           P    P    3    3    5        9    9                 *
0044            *           P    P         3    5        9    9                 *
0045            *           PPPPP       3333     555     99999                  *
0046            *           P              3        5        9                   *
0047            *           P          3   3    5   5    9    9                 *
0048            *           P           3333     555     9999                   *
0049            *                                                                *
0050            ********************************************************************
```

```
0055                    DEF   CRUNCH
0056 0000
0057                    REF   BUFLEV,CRNBUF,FAC,CHAT,VARW,VARW1,VARA,RAMPTR
0058                    REF   KEYTAB,CRNEND,GETSTK,PUTSTK,PRGFLG,VAR5,ARG
0059                    REF   R0LB,R1LB,R2LB,R6LB,R8LB,R9LB,C4,ERRCOD,ARG2
0060                    REF   VRAM,GROM,GETV1,GETNB,GETCHR,ARG4
0061                    REF   VDPWD,VDPRD,WRVDP,GRMWA
0062                    REF   LBCPMO,CPTBL
0063 0000
0064       0022  QUOTE  EQU   >22
0065       002C  COMMA  EQU   >2C
0066 0000
0067       0002  LIST$  EQU   >02
0068       0005  OLD$   EQU   >05
0069       0007  SAVE$  EQU   >07
0070       0008  MERGE$ EQU   >08
0071       0081  ELSE$  EQU   >81
0072       0082  SSEP$  EQU   >82
0073       0083  TREM$  EQU   >83
0074       0085  GO$    EQU   >85
0075       0086  GOTO$  EQU   >86
0076       0087  GOSUB$ EQU   >87
0077       0088  RETUR$ EQU   >88
0078       008E  BREAK$ EQU   >8E
0079       008F  UNBRK$ EQU   >8F
0080       0093  DATA$  EQU   >93
0081       0094  RESTO$ EQU   >94
0082       009A  REM$   EQU   >9A
0083       009D  CALL$  EQU   >9D
0084       00A1  SUB$   EQU   >A1
0085       00A3  IMAGE$ EQU   >A3
0086       00A5  ERROR$ EQU   >A5
0087       00A9  RUN$   EQU   >A9
0088       00B0  THEN$  EQU   >B0
0089       00B1  TO$    EQU   >B1
0090       00B3  COMMA$ EQU   >B3
0091       00B5  COLON$ EQU   >B5
0092       00C7  QUOTE$ EQU   >C7
0093       00C8  UNQST$ EQU   >C8
0094       00C9  LN$    EQU   >C9
0095       00ED  USING$ EQU   >ED
0096 0000
0097       000A  MAXKEY EQU   10
0098             *     Equates for use with the Character Property Table
0099       0020  LLC    EQU   >20
0100 0000
0101       0000  CPNIL  EQU   >00          " $ % ' ?
0102       0002  CPDIG  EQU   >02          digit (0-9)
0103       0004  CPNUM  EQU   >04          digit, period, E
0104       0008  CPOP   EQU   >08          1 char operators( !#&*+-/<=>^ )
0105       0010  CPMO   EQU   >10          multiple operator( : )
0106       0020  CPALPH EQU   >20          A-Z, @, _
0107       0040  CPBRK  EQU   >40          ( ) , ;
0108       0080  CPSEP  EQU   >80          space
0109       0022  CPALNM EQU   CPALPH+CPDIG alpha+digit - legal variable chars
```

```
0111                *       CRUNCH COPIES A LINE (NORMALLY IN LINEBF) TO CRNBUF
0112                *       IN THE PROCESS, IT TURNS THE LINE NUMBER (IF ANY) T
0113                *       BINARY, AND CONVERTS ALL RESERVED WORDS TO TOKENS.
0114                *       CALL IS BY GPL XML FOLLOWED BY A SINGLE BYTE WHICH
0115                *       INDICATES THE TYPE OF CRUNCH TO BE DONE.  POSSIBLE
0116                *       TYPES INCLUDE:
0117                *            00 - Normal crunch
0118                *            01 - crunch as a data statement(input stmt)
0119                *
0120                *       REGISTERS:
0121                *       R0-R1/ SCRATCH
0122                *       R2-R3/ SCRATCH
0123                *       R4/ POINTS TO R8LB
0124                *       R5/ POINTS TO LENGTH BYTE OF STRING/NUMERIC
0125                *       R6/ INDICATES NUMERIC COPY MODE (NUMERIC/LINE #)
0126                *       R7/ MODE OF COPY (STRINGS, NAMES, REMS, ETC)
0127                *       R8/ CHARACTER BUFFER
0128                *       R9/ POINTS TO NAME DURING KEYWORD SCAN
0129                *       R11-R12/ LINKS
0130                *       R13/ GROM READ DATA POINTER
0131                *       R15/ VDP WRITE ADDRESS POINTER
```

```
0133 0000 C30B  CRUNCH MOV  R11,R12          Save return link
0134                   *-------------------CONDITIONAL ASSEMBLY--------------------*
0135                          ASMIF VERS=DX10
0136                          MOVB *R15+,R3          Read call code
0137                          ASMELS
0138 0002 D0DD                MOVB *R13,R3           Read call code
0139                          ASMEND
0140                   *-----------------END OF CONDITIONAL ASSEMBLY---------------*
0141 0004 06A0                BL   @PUTSTK           Save GROM address
     0006 0000
0142 0008 04E0                CLR  @FAC              Assume no line number
     000A 0000
0143 000C 0204                LI   R4,R8LB           Set up W/S low-byte pointer
     000E 0000
0144 0010 04C8                CLR  R8                Initialize character buffer
0145 0012 06A0                BL   @GETNB            Scan line for 1st good char
     0014 0000
0146 0016 D501                MOVB R1,*R4            Save character
0147 0018 1370                JEQ  CRU28             If empty line - return
0148                   *      Now check crunch call mode - normal or input statement
0149 001A 0983                SRL  R3,8              Normal crunch call?
0150 001C 1307                JEQ  CRU01             Yes-crunch the statement
0151                   *      Initialize for input statement crunch
0152 001E 0202                LI   R2,CRU84          No-must be crunch input stmt
     0020 02BA'
0153 0022 020A                LI   R10,CRU83             so set up mode indicators
     0024 0288'
0154 0026 0207                LI   R7,CRU80
     0028 02DA'
0155 002A 101F                JMP  CRU10             And jump into it
0156 002C
0157                   *      Initialize for normal line crunch
0158 002C 05A0  CRU01  INC  @BUFLEV          Indicate CRNBUF is destroyed
     002E 0000
0159 0030 04E0                CLR  @ARG4             Assume not symbol
     0032 0000
0160 0034 D808                MOVB R8,@PRGFLG        Clear program flag
     0036 0000
0161 0038 06A0                BL   @GETINT           Try to read a line number
     003A 03FE'
0162 003C C800                MOV  R0,@FAC           Put line number into final
     003E 000A'
0163 0040 1304                JEQ  CRU02             If no line number
0164 0042 06A0                BL   @GETNB            Skip all leading spaces
     0044 0014'
0165 0046 D501                MOVB R1,*R4            Save character in R8LB
0166 0048 1358                JEQ  CRU28             If nothing left in line
0167 004A 0207  CRU02  LI   R7,CRU16          Set normal scan mode
     004C 008A'
0168 004E 0206                LI   R6,CRU96          Set normal numeric scan mode
     0050 032C'
0169 0052 100B                JMP  CRU10             Merge into normal scan code
0170 0054
0171                   *      Main loop of the input copy routine. Sets R8LB to
0172                   *      next character, R0 to its character property byte
0173                   *      R7 indicates dispatch mode.
0174 0054
0175 0054 0206  CRU04  LI   R6,CRU96          Set normal numeric mode
     0056 032C'
0176 0058 0207  CRU05  LI   R7,CRU16          Set normal scan mode
```

```
      005A 008A'
0177  005C 06A0  CRU06  BL    @PUTCHR           Copy into crunch buffer
      005E 03C6'
0178  0060 06A0  CRU08  BL    @GETCHR           Get next input character
      0062 0000
0179  0064 04C0        CLR   R0                Assume nil property
0180  0066 D501        MOVB  R1,*R4            Copy to crunch buffer
0181  0068 1302        JEQ   CRU12             Finish up if we reach a null
0182  006A D028  CRU10  MOVB  @CPTBL(R8),R0     Fetch char's prop table val
      006C 0000
0183  006E 0457  CRU12  B     *R7               Dispatch to appropriate code
0184  0070 C208  CRU14  MOV   R8,R8            End of line?
0185  0072 16F4        JNE   CRU06             Not yet
0186  0074 C0E0  CRU15  MOV   @RAMPTR,R3        Now check for trailing spaces
      0076 0000
0187  0078 0603        DEC   R3                Backup to read last char
0188  007A 06A0        BL    @GETV1            Go read it
      007C 0000
0189  007E 9801        CB    R1,@CBH20         Last char a space?
      0080 01CB'
0190  0082 163B        JNE   CRU28             No,so end of line-exit
0191  0084 0620        DEC   @RAMPTR           Yes, backup pointer to delete
      0086 0076'
0192  0088 10F5        JMP   CRU15             And test new last-character
```

```
0194                    *       Normal scan mode--figures out what to do with this
0195                    *       character
0196 008A D514  CRU16   MOVB    *R4,*R4             At end of line?
0197 008C 1336          JEQ     CRU28               Yes - clean up and return
0198 008E D000          MOVB    R0,R0               Set condition on char prop
0199 0090 11E7          JLT     CRU08               Ignore separators (spaces)
0200 0092 C260          MOV     @RAMPTR,R9          Save crunch pointer
     0094 0086'
0201 0096 0A20          SLA     R0,2                Scan property bits 1 and 2
0202 0098 183E          JOC     CRU32               Break chars are 1 char tokens
0203 009A 110F          JLT     CRU18               Alpha - prepare to pack name
0204 009C 0A20          SLA     R0,2                Scan property bits 3 and 4
0205 009E 1710          JNC     CRU20               Jump if not multi-char oper
0206 00A0 06A0          BL      @GETCHR             Check next char to see if we
     00A2 0062'
0207 00A4 0981          SRL     R1,8                   have a 2-char operator
0208 00A6 1337          JEQ     CRU32               If read end of line-single op
0209 00A8 06A0          BL      @BACKUP             Backup read pointer
     00AA 03AC'
0210 00AC 9821          CB      @CPTBL(R1),@LBCPMO  Next char also a multi-op?
     00AE 006C'
     00B0 0000
0211 00B2 1631          JNE     CRU32               No--must want single-char op
0212 00B4 06A0          BL      @PUTCHR             Copy in first char of op
     00B6 03C6'
0213 00B8 1030          JMP     CRU36               And scan keyword table
0214 00BA
0215                    *       Set name copy mode
0216 00BA 0207  CRU18   LI      R7,CRU76            Alphabetic:set name copy mode
     00BC 026C'
0217 00BE 10CE          JMP     CRU06               And resume copy
0218 00C0
0219                    *       Handle single character operators
0220 00C0 112A  CRU20   JLT     CRU32               Bit 4: single character oper
0221 00C2 0A20          SLA     R0,2                Scan property bits 5 and 6
0222 00C4 180E          JOC     CRU24               If numeric
0223 00C6 1112          JLT     CRU26               If digit only
0224 00C8 0288          CI      R8,QUOTE            Is it a string quote?
     00CA 0022
0225 00CC 1678          JNE     ERRIVN              No--unknown char so error
0226 00CE C287          MOV     R7,R10              Yes--save current mode
0227 00D0 0208  CRU22   LI      R8,QUOTE$           Convert char to quote token
     00D2 00C7
0228 00D4 06A0          BL      @PUTCHR             Put in token
     00D6 03C6'
0229 00D8 0207          LI      R7,CRU68            Set string-copy mode
     00DA 0248'
0230 00DC C160          MOV     @RAMPTR,R5          Save ptr to length byte
     00DE 0094'
0231 00E0 10BD          JMP     CRU06               Continue copy w/qutoe token
0232 00E2 0288  CRU24   CI      R8,'.'              A decimal point?
     00E4 002E
0233 00E6 1602          JNE     CRU26               No-decode as numeric/line #
0234 00E8 0206          LI      R6,CRU96            Yes-decode as numeric
     00EA 032C'
0235 00EC 0456  CRU26   B       *R6                 Handle numeric or line #
0236 00EE
0237 00EE 0460  BERRSY  B       @ERRSYN             Long distance SYNTAX ERROR
     00F0 03A4'
```

```
0239 00F2 06A0  CRU27   BL    @PUTCHR           Put out last char before end
     00F4 03C6'
0240 00F6 05A0          INC   @VARW             Skip last character
     00F8 0000
0241              *     Here for successful completion of scan
0242 00FA 06C8  CRU28   SWPB  R8                Mark end of line with a null
0243 00FC 06A0          BL    @PUTCHR           Put the end of line in
     00FE 03C6'
0244      0102' CRNADD  EQU   $+2
0245 0100 0200          LI    R0,CRNBUF         Get start of crunch buffer
     0102 0000
0246 0104 0500          NEG   R0                Negate for backwards add
0247 0106 A020          A     @RAMPTR,R0        Calculate line length
     0108 00DE'
0248 010A D820          MOVB  @R0LB,@CHAT       Save length for GPL
     010C 0000
     010E 0000
0249 0110 06A0          BL    @GETSTK           Restore GROM address
     0112 0000
0250 0114 045C          B     *R12              Return with ptr beyond null
```

```
0252                    *       Keyword table scanning routine. Name has already been
0253                    *       copied into crunch area starting at R9; RAMPTR points
0254                    *       just beyond name in input line.
0255                    *       R3 is name length, R1 indexes into the table.
0256 0116 06A0  CRU32   BL    @BACKUP           Fix ptr for copy(next line)
     0118 03AC'
0257 011A 06A0  CRU36   BL    @GETCHR           Read last character
     011C 00A2'
0258 011E D501          MOVB  R1,*R4            Put into output buffer
0259 0120 06A0          BL    @PUTCHR           Copy into crunch buffer
     0122 03C6'
0260 0124
0261 0124 C0E0  CRU38   MOV   @RAMPTR,R3        Get end pointer
     0126 0108'
0262 0128 60C9          S     R9,R3             Sub start to get len of name
0263 012A 0283          CI    R3,MAXKEY         Is longer than any keyword?
     012C 000A
0264 012E 1B71          JH    CRU61             Yes-can't be a keyword
0265 0130 C083          MOV   R3,R2             Get name length and
0266 0132 0602          DEC   R2                Correct 0-length name indexing
0267 0134 0A12          SLA   R2,1              Turn it into an index
0268                    *-----------------CONDITIONAL ASSEMBLY---------------------
0269                            ASMIF VERS=DX10
0270
0271                            MOV   R2,R15            Put index in GROM read reg
0272                            AI    R15,KEYTAB        Add in address of table list
0273                            AI    R15,GROM          Add in GROM offset
0274                            MOVB  *R15+,R2          Read address of correct table
0275                            MOVB  *R15,@R2LB        Both bytes
0276                    *       R2 now contains the address of the correct table
0277                    CRU40   MOV   R2,R15            Load address of table
0278                            MOV   R3,R0             Copy of length for compare
0279                            AI    R15,GROM          Add in GROM offset
0280                            A     R3,R2             Address of next keyword in tab
0281                            INC   R2                Skip token value
0282                            MOV   R9,R14            Source is in VDP
0283                            AI    R14,VRAM
0284                    CRU42   CB    *R14+,*R15+       Check next character
0285                            JL    CRU61A            If no match possible
0286                            JNE   CRU40             Didn't match - try next one
0287                            DEC   R0                Compared all?
0288                            JNE   CRU42             No - check next one
0289                    CRU46   MOV   R9,@RAMPTR        Name matched so throw out name
0290                            MOVB  *R15,*R4          Read the token value
0291
0292                            ASMELS
0293 0136
0294 0136 0222                  AI    R2,KEYTAB         Add in address of table list
     0138 0000
0295 013A DB42                  MOVB  R2,@GRMWA(R13)    Load address to GROM
     013C 0000
0296 013E 06C2                  SWPB  R2
0297 0140 DB42                  MOVB  R2,@GRMWA(R13)
     0142 013C'
0298 0144 D09D                  MOVB  *R13,R2           Read address of correct table
0299 0146 D81D                  MOVB  *R13,@R2LB        Both bytes
     0148 0000
0300                    *       R2 now contains the address of the correct table
0301 014A DB42  CRU40   MOVB  R2,@GRMWA(R13)    Load address of table
     014C 0142'
```

```
0302 014E C003          MOV  R3,R0              Copy of length for compare
0303 0150 DB60          MOVB @R2LB,@GRMWA(R13)
     0152 0148'
     0154 014C'
0304 0156 D7E0          MOVB @R9LB,*R15         Source is in VDP
     0158 0000
0305 015A A083          A    R3,R2              Addr of next keyword in table
0306 015C D7C9          MOVB R9,*R15
0307 015E 0582          INC  R2                 Skip token value
0308 0160 9760  CRU42   CB   @VDPRD,*R13        Compare the character
     0162 0000
0309 0164 1A59          JL   CRU61A             If no match possible
0310 0166 16F1          JNE  CRU40              No match-but match possible
0311 0168 0600          DEC  R0                 Compared all?
0312 016A 16FA          JNE  CRU42              No - check next one
0313 016C C809          MOV  R9,@RAMPTR         Name matched so throw out name
     016E 0126'
0314 0170 D51D          MOVB *R13,*R4           Read the token value
0315 0172
0316                    ASMEND
0317                *---------------END OF CONDITIONAL ASSEMBLY----------------*
0318 0172 04E0          CLR  @ARG4              Indicate keyword found
     0174 0032'
```

```
0320                    *       Check for specially crunched statements
0321 0176 0207                  LI    R7,CRU14          Assume a rem statement
     0178 0070'
0322 017A 0200                  LI    R0,SPECTB-1       Now check for special cases
     017C 0447'
0323 017E 0288                  CI    R8,MERGE$         Is this a command?
     0180 0008
0324 0182 1B06                  JH    CRU47             No-continue on
0325 0184 C0E0                  MOV   @FAC,R3           Yes-attempt to put in program?
     0186 003E'
0326 0188 161C                  JNE   ERRCIP            Yes - *COMMAND ILLEGAL IN PROG
0327 018A 0289                  CI    R9,CRNBUF         Command 1st token in line?
     018C 0102'
0328 018E 16AF                  JNE   BERRSY            No - *SYNTAX ERROR
0329 0190 0580  CRU47   INC     R0                      Skip offset value
0330 0192 9C14                  CB    *R4,*R0+          In special table?
0331 0194 1320                  JEQ   CRU53A            Yes - handle it
0332 0196 1BFC                  JH    CRU47             If still possible match
0333 0198 0288                  CI    R8,MERGE$         A specially scanned command?
     019A 0008
0334 019C 1AAA                  JL    CRU27             Yes-exit crunch
0335 019E 0200                  LI    R0,LNTAB          Now check for line numbers
     01A0 043A'
0336 01A2 9C14  CRU48   CB      *R4,*R0+                In table?
0337 01A4 1309                  JEQ   CRU52             Yes - change to line # crunch
0338 01A6 1BFD                  JH    CRU48             May still be in table
0339 01A8 0288                  CI    R8,COMMA$         Just crunch a comma?
     01AA 00B3
0340 01AC 1303                  JEQ   CRU50             Yes-so retain current numeric
0341 01AE 0288                  CI    R8,TO$            Just crunch a TO?
     01B0 00B1
0342 01B2 160F                  JNE   CRU53             No-so reset to normal numeric
0343 01B4 0460  CRU50   B       @CRU05                  Yes-resume normal copy
     01B6 0058'
0344 01B8 0206  CRU52   LI      R6,CRU100               Set line number scan mode
     01BA 0360'
0345 01BC 10FB                  JMP   CRU50             Set normal scan mode
0346 01BE
0347 01BE
0348 01BE 05A0  ERRIVN  INC     @ERRCOD                 *ILLEGAL VARIABLE NAME        7
     01C0 0000
0349 01C2 05A0  ERRCIP  INC     @ERRCOD                 *COMMAND ILLEGAL IN PROGRAM   6
     01C4 01C0'
0350 01C6 05A0  ERRNQT  INC     @ERRCOD                 *NONTERMINATED QUOTED STRING  5
     01C8 01C4'
0351       01CB' CBH20  EQU     $+1
0352 01CA A820  ERRNTL  A       @C4,@ERRCOD             *NAME TOO LONG                4
     01CC 0000
     01CE 01C8'
0353 01D0 1094                  JMP   CRU28             Exit back to GPL
0354       01D2' OFFSET EQU     $
0355 01D2 0460  CRU53   B       @CRU04                  Stmt sep resets to normal scan
     01D4 0054'
0356 01D6
0357 01D6 D050  CRU53A  MOVB    *R0,R1                  Pick up offset from table
0358 01D8 0981          SRL     R1,8                    Make into offset
0359 01DA 0461          B       @OFFSET(R1)             Goto special case handler
     01DC 01D2'
0360 01DE
```

```
0362                    *       Process a LIST statement
0363 01DE 06A0  CRU57   BL      @PUTCHR             Put the list token in
     01E0 03C6'
0364 01E2 06A0          BL      @GETNB              Get next character
     01E4 0044'
0365 01E6 0281          CI      R1,QUOTE*256        Device name available?
     01E8 2200
0366 01EA 1687          JNE     CRU28               No-no more to crunch-exit
0367 01EC 020A          LI      R10,CRU106          Yes-set after string scan mode
     01EE 0384'
0368 01F0 0460          B       @CRU22              Crunch the device name
     01F2 00D0'
0369 01F4
0370                    *       Process an IMAGE statement
0371 01F4 020A  CRU54   LI      R10,CRU83B          Image after-string copy mode
     01F6 028E'
0372 01F8 1002          JMP     CRU59               Handle similar to data stmt
0373 01FA
0374                    *       Process a DATA statement
0375 01FA 020A  CRU58   LI      R10,CRU83           After-datum skip spaces
     01FC 0288'
0376 01FE 8820  CRU59   C       @RAMPTR,@CRNADD     Image&data must be 1st on line
     0200 016E'
     0202 0102'
0377 0204 1648          JNE     JNESY1              If not - error
0378 0206 0202          LI      R2,CRU84            (non-)quote string copy mode
     0208 02BA'
0379 020A 0207  CRU60   LI      R7,CRU80            Now set check-for-quote mode
     020C 02DA'
0380 020E 0460  CRU74   B       @CRU06              And copy in statement token
     0210 005C'
0381 0212
0382                    *       Here when don't find something in the keyword table
0383 0212 0283  CRU61   CI      R3,15               Is it longer than name can be?
     0214 000F
0384 0216 1BD9          JH      ERRNTL              Yes-name too long
0385 0218 C020  CRU61A  MOV     @ARG4,R0            Symbol name last time too?
     021A 0174'
0386 021C 163C          JNE     JNESY1              Yes-can't have 2 in a row
0387 021E 0620          DEC     @ARG4               Indicate symbol now
     0220 021A'
0388 0222 0207  CRU62   LI      R7,CRU16            No keyword; leave in CRNBUF
     0224 008A'
0389 0226 0206          LI      R6,CRU96            Assume normal numeric scan
     0228 032C'
0390 022A 0460  CRU64   B       @CRU08              And continue to scan line
     022C 0060'
0391 022E
0392                    *       Process a SUB statement
0393 022E C0E0  CRU65   MOV     @RAMPTR,R3          Get the current crunch ptr
     0230 0200'
0394 0232 0603          DEC     R3                  Point at last char put in
0395 0234 06A0          BL      @GETV1              Read it
     0236 007C'
0396 0238 9801          CB      R1,@GO$TOK          Was it a GO?
     023A 043B'
0397 023C 13BD          JEQ     CRU52               Yes-SUB is part of gosub
0398                    *       Process a CALL or SUB statement
0399 023E 0207  CRU66   LI      R7,CRU93            Set name copy
     0240 031A'
```

```
0400 0242 10E5          JMP   CRU74             And get next character
0401 0244 0460  CRU32L  B     @CRU32
     0246 0116'
```

```
0403                    *       Now the various mode copy routines:  string, names
0404                    *       image and data statements
0405                    *Strings
0406 0248 C208  CRU68   MOV   R8,R8             Premature end of line?
0407 024A 13BD          JEQ   ERRNQT            Yes-*NONTERMINATED QUOTED STR
0408 024C 0288          CI    R8,QUOTE          Reach end of string?
     024E 0022
0409 0250 16DE          JNE   CRU74             No--continue copying
0410 0252 06A0          BL    @GETCHR           Get next character
     0254 011C'
0411 0256 D041          MOVB  R1,R1             Read end of line?
0412 0258 1305          JEQ   CRU70             Yes-can't be double quote
0413 025A 0281          CI    R1,QUOTE*256      Is it two quotes in a row?
     025C 2200
0414 025E 13D7          JEQ   CRU74             Yes-copy in a normal quote
0415 0260 06A0          BL    @BACKUP           No-backup & rtn to normal scan
     0262 03AC'
0416 0264 C1CA  CRU70   MOV   R10,R7            Needed for image/data stmts
0417 0266 06A0  CRU72   BL    @LENGTH           Calculate length of string
     0268 03E4'
0418 026A 10DF          JMP   CRU64             Resume scan
0419 026C
0420 026C
0421                    *Names
0422 026C 0240  CRU76   ANDI  R0,CPALNM*256     Is this char alpha or a digit
     026E 2200
0423 0270 16CE          JNE   CRU74             Yes- continue packing
0424                    *                       No--finished w/ name packing
0425 0272 0288          CI    R8,'$'            Does name end with a $?
     0274 0024
0426 0276 13E6          JEQ   CRU32L            Yes-include it in name
0427 0278 D514          MOVB  *R4,*R4           At an end of line?
0428 027A 1302          JEQ   CRU79             Yes-don't back up pointer
0429 027C 06A0          BL    @BACKUP           Backup for next char
     027E 03AC'
0430 0280 0460  CRU79   B     @CRU38            Jump to name/keyword check
     0282 0124'
0431 0284 0460  CRU82   B     @CRU22
     0286 00D0'
```

```
0433                    *DATA: Scan spaces after a quoted string datum
0434 0288 0288  CRU83   CI    R8,COMMA          Hit a comma?
     028A 002C
0435 028C 1321          JEQ   CRU85A            Yes-get back into scan
0436                    *IMAGE: Scan spaces after a quoted string datum
0437 028E D000  CRU83B  MOVB  R0,R0             At a space?
0438 0290 11CC          JLT   CRU64             Yes - ignore it
0439 0292 C208          MOV   R8,R8             At end of line?
0440 0294 13C6          JEQ   CRU62             Yes-exit scan
0441 0296 1060  JNESY1  JMP   JNESYN            No-unknown character
0442 0298
0443                    *DATA: Scan imbedded blanks and check trailing blanks
0444 0298 C820  CRU83A  MOV   @VARW,@ARG2       Save input pointer
     029A 00F8'
     029C 0000
0445 029E 06A0          BL    @GETNB            Look for next non-blank
     02A0 01E4'
0446 02A2 D041          MOVB  R1,R1             At end of line?
0447 02A4 1337          JEQ   CRU92             Yes-end string and exit
0448 02A6 028A          CI    R10,CRU83B        Scanning an image?
     02A8 028E'
0449 02AA 1303          JEQ   CRU83C            Yes-commas are not significant
0450 02AC 0281          CI    R1,COMMA*256      Hit a comma?
     02AE 2C00
0451 02B0 130D          JEQ   CRU85             Yes-ignore trailing spaces
0452 02B2 C820  CRU83C  MOV   @ARG2,@VARW       No-restore input pointer
     02B4 029C'
     02B6 029A'
0453 02B8 10AA          JMP   CRU74                 and include imbedded space
0454 02BA
0455                    *DATA: Scan unquoted strings
0456 02BA 11EE  CRU84   JLT   CRU83A            If hit a space-end of string
0457 02BC C208          MOV   R8,R8             At end-of-line?
0458 02BE 132A          JEQ   CRU92             Yes-put in length and exit
0459 02C0 0288          CI    R8,COMMA          Reached a comma?
     02C2 002C
0460 02C4 16A4          JNE   CRU74             No-scan unquoted string
0461 02C6 028A          CI    R10,CRU83B        Scanning an image stmt?
     02C8 028E'
0462 02CA 13A1          JEQ   CRU74             Commas are not significant
0463 02CC 06A0  CRU85   BL    @LENGTH           Yes-end the string
     02CE 03E4'
0464 02D0 0208  CRU85A  LI    R8,COMMA$         Load a comma token
     02D2 00B3
0465 02D4 05A0          INC   @VAR5             Count comma for input stmt
     02D6 0000
0466 02D8 1098          JMP   CRU60             And resume in string mode
0467 02DA
0468                    *IMAGE/DATA: Check for leading quote mark
0469 02DA 11A7  CRU80   JLT   CRU64             Ignore leading separators
0470 02DC 0288          CI    R8,QUOTE          Quoted string?
     02DE 0022
0471 02E0 13D1          JEQ   CRU82             Yes-like any string, R10 OK
0472 02E2 C208          MOV   R8,R8             End of line?
0473 02E4 1361          JEQ   BCRU28            Yes-end it
0474 02E6 028A          CI    R10,CRU83B        Scanning an image?
     02E8 028E'
0475 02EA 1303          JEQ   CRU88             Yes ignore commas
0476 02EC 0288          CI    R8,COMMA          At a comma?
     02EE 002C
```

```
0477 02F0 13EF          JEQ  CRU85A          Yes-put it in directly
0478 02F2 C1C2  CRU88   MOV  R2,R7           No--set unquote str copy mod
0479 02F4
0480                *IMAGE & DATA: Scan unquoted strings
0481 02F4 0208  CRU86   LI   R8,UNQST$       Load unquoted string token
     02F6 00C8
0482 02F8 06A0          BL   @PUTCHR         Put the token in
     02FA 03C6'
0483 02FC C160          MOV  @RAMPTR,R5      Save current crunch pointer
     02FE 0230'
0484 0300 06A0          BL   @BACKUP         Back up to scan again
     0302 03AC'
0485 0304 1084  CRU87   JMP  CRU74           Resume scan
```

```
0487                    *CALL and SUB statements
0488 0306 0240  CRU94   ANDI RO,CPALNM*256      Still an alpha-numeric?
     0308 2200
0489 030A 1681          JNE  CRU74              Yes-include in name
0490 030C C208          MOV  R8,R8              At end of line?
0491 030E 1302          JEQ  CRU92              Yes-get out now
0492 0310 06A0  CRU90   BL   @BACKUP            Yes-reset read pointer
     0312 03AC'
0493 0314 0207  CRU92   LI   R7,CRU16           Normal scanning mode
     0316 008A'
0494 0318 10A6          JMP  CRU72              Calculate & put in string len
0495 031A
0496                    *CALL and SUB statements before hit name
0497 031A 1187  CRU93   JLT  CRU64              If a space, ignore it
0498 031C C000          MOV  RO,RO              Premature EOL or NIL char-prop?
0499 031E 1342          JEQ  ERRSYN             Yes- *SYNTAX ERROR
0500 0320 0240          ANDI RO,CPALPH*256      An aplhabetic to start name?
     0322 2000
0501 0324 133F          JEQ  ERRSYN             No-syntax error
0502 0326 0207          LI   R7,CRU94           Set up to copy name
     0328 0306'
0503 032A 10E4          JMP  CRU86              Put in the unqst token
0504 032C
0505                    *Numerics
0506 032C 0207  CRU96   LI   R7,CRU98           Set after-initialize scan
     032E 0336'
0507 0330 04E0          CLR  @ARG               Clear the 'E' flag
     0332 0000
0508 0334 10DF          JMP  CRU86              Set up for the numeric
0509 0336 C208  CRU98   MOV  R8,R8              At end of line?
0510 0338 13ED          JEQ  CRU92              Yes end the number
0511 033A 0A20          SLA  RO,2               Scan property bit 2
0512 033C 1108          JLT  CRU99A             If alpha-might be 'E'
0513 033E 0A30          SLA  RO,3               Scan property bits 4 and 5
0514 0340 1702          JNC  CRU99              Bit 4=oper - if not oper-jmp
0515 0342 C020          MOV  @ARG,RO            If operator-follow an 'E'?
     0344 0332'
0516 0346 04E0  CRU99   CLR  @ARG               Previous char no longer an 'E'
     0348 0344'
0517 034A 11DC          JLT  CRU87              If still numeric
0518 034C 10E1          JMP  CRU90              No longer numeric
0519 034E 0288  CRU99A  CI   R8,'E'             'E' to indicate an exponent?
     0350 0045
0520 0352 16DE          JNE  CRU90              No-so end the numeric
0521 0354 C020          MOV  @ARG,RO            An 'E' already encountered?
     0356 0348'
0522 0358 1625  JNESYN  JNE  ERRSYN             Yes-so error
0523 035A 0720          SETO @ARG               No-indicated 1 encountered now
     035C 0356'
0524 035E 10D2          JMP  CRU87              And include it in the number
0525 0360
0526                    *Line numbers
0527 0360 C208  CRU100  MOV  R8,R8              At end of line?
0528 0362 1322          JEQ  BCRU28             Yes - exit crunch
0529 0364 06A0          BL   @GETINT            Try to get a line number
     0366 03FE'
0530 0368 C000          MOV  RO,RO              Get a line number?
0531 036A 130A          JEQ  CRU105             No-back to normal numeric mode
0532 036C 0208          LI   R8,LN$             Load a line number token
     036E 00C9
```

```
0533 0370 06A0          BL    @PUTCHR           Put it out
     0372 03C6'
0534 0374 C200          MOV   R0,R8             Set up to put out binary #
0535 0376 06C8          SWPB  R8                Swap to put MSByte of # 1st
0536 0378 06A0          BL    @PUTCHR           Put out 1st byte of line #
     037A 03C6'
0537 037C 0988          SRL   R8,8              Bare the 2nd byte of line #
0538 037E 10C2          JMP   CRU87             Jump back into it
0539 0380
0540 0380 0460  CRU105  B     @CRU04            Back to normal numeric mode
     0382 0054'
0541 0384
0542                    *Handle a list statement
0543 0384 11CA  CRU106  JLT   CRU93             If space-ignore it
0544 0386 C208          MOV   R8,R8             At the end of line?
0545 0388 130F          JEQ   BCRU28            Yes - exit crunch
0546 038A 0288          CI    R8,':'            Get a colon?
     038C 003A
0547 038E 160A          JNE   ERRSYN            No - *SYNTAX ERROR
0548 0390 0208          LI    R8,COLON$         Need to put colon in
     0392 00B5
0549 0394 0460          B     @CRU27            And exit crunch
     0396 00F2'
```

```
0551                    *       Error handling routine
0552 0398 05A0  ERRLTL  INC   @ERRCOD           * LINE TOO LONG    3
     039A 01CE'
0553 039C 0660          DECT  @RAMPTR           Backup so can exit to GPL
     039E 02FE'
0554 03A0
0555 03A0 05A0  ERRBLN  INC   @ERRCOD           * BAD LINE NUMBER  2
     03A2 039A'
0556 03A4
0557 03A4 05A0  ERRSYN  INC   @ERRCOD           * SYNTAX ERROR     1
     03A6 03A2'
0558 03A8
0559 03A8 0460  BCRU28  B     @CRU28            Exit back to GPL
     03AA 00FA'
0560 03AC
0561 03AC
0562 03AC
0563 03AC
0564 03AC
0565                    *       Back up pointer in input line to rescan last char
0566 03AC 0620  BACKUP  DEC   @VARW             Back up the pointer
     03AE 02B6'
0567                    *-----------------CONDITIONAL ASSEMBLY---------------------*
0568                            ASMIF VERS=DX10
0569
0570                            MOV   @VARW,R14         Move to R14 for read
0571                            AI    R14,VRAM          Add in VDP offset
0572                            LI    R0,>7F00          >7F is an edge character
0573                            SB    *R14,R0           At an edge character?
0574
0575                            ASMELS
0576 03B0
0577 03B0 D7E0                  MOVB  @VARW1,*R15       Write LSByte of address
     03B2 0000
0578 03B4 1000                  NOP
0579 03B6 D7E0                  MOVB  @VARW,*R15        Write MSByte of address
     03B8 03AE'
0580 03BA 0200                  LI    R0,>7F00          >7F is an edge character
     03BC 7F00
0581 03BE 7020                  SB    @VDPRD,R0         At an edge character?
     03C0 0162'
0582 03C2
0583                            ASMEND
0584                    *---------------END OF CONDITIONAL ASSEMBLY----------------*
0585 03C2 13F4                  JEQ   BACKUP            Yes - back up one more
0586 03C4 045B                  RT                      And return to caller
```

```
0588                    *       Put a character into the crunch buffer
0589 03C6 C060  PUTCHR  MOV   @RAMPTR,R1        Fetch the current pointer
     03C8 039E'
0590 03CA 0281          CI    R1,CRNEND         At end of buffer?
     03CC 0000
0591 03CE 1BE4          JH    ERRLTL            Yes - LINE TOO LONG
0592                    *-------------------CONDITIONAL ASSEMBLY--------------------*
0593                    ASMIF VERS=DX10
0594                    MOV   R1,R14            Use R14 as usual on /10
0595                    AI    R14,VRAM          Add in VDP RAM offset
0596                    MOVB  *R4,*R14          Put the byte there
0597                    INC   @RAMPTR           Increment the pointer
0598
0599                    ASMELS
0600 03D0
0601 03D0 D7E0          MOVB  @R1LB,*R15        Put out LSByte of address
     03D2 0000
0602 03D4 0261          ORI   R1,WRVDP          Enable VDP write
     03D6 0000
0603 03D8 D7C1          MOVB  R1,*R15           Put out MSByte of address
0604 03DA 05A0          INC   @RAMPTR           Increment the pointer
     03DC 03C8'
0605 03DE D814          MOVB  *R4,@VDPWD        Write out the byte
     03E0 0000
0606                    ASMEND
0607                    *-------------------END OF CONDITIONAL ASSEMBLY-------------*
0608 03E2 045B          RT                      And return
0609 03E4
0610                    *       Calculate and put length of string/number into
0611                    *       length byte
0612 03E4 C0CB  LENGTH  MOV   R11,R3            Save return address
0613 03E6 C020          MOV   @RAMPTR,R0        Save current crunch pointer
     03E8 03DC'
0614 03EA C200          MOV   R0,R8             Put into R8 for PUTCHR below
0615 03EC 6205          S     R5,R8             Calculate length of string
0616 03EE 0608          DEC   R8                RAMPTR is post-incremented
0617 03F0 C805          MOV   R5,@RAMPTR        Address of length byte
     03F2 03E8'
0618 03F4 06A0          BL    @PUTCHR           Put the length in
     03F6 03C6'
0619 03F8 C800          MOV   R0,@RAMPTR        Restore crunch pointer
     03FA 03F2'
0620 03FC 0453          B     *R3               And return
```

```
0622                    *               GET A SMALL NON-NEGATIVE INTEGER
0623                    *       CALL: VARW/ TEXT POINTER, POINTS TO SECOND CHARACTER
0624                    *       R8/ FIRST CHARACTER IN LOW BYTE
0625                    *       BL   @GETINT
0626                    *       RO/ NUMBER
0627                    *       VARW/ TEXT POINTER,
0628                    *       IF THERE IS A NUMBER, POINTS TO CHARACTER AFTER NUMBER
0629                    *       IF THERE IS NOT A NUMBER, UNCHANGED.
0630                    *       R8/ 0 IN HIGH BYTE
0631                    *       DESTROYS: R1, R2
0632 03FE C0CB  GETINT MOV   R11,R3             Save return address
0633 0400 C008          MOV   R8,RO             Get possible digit
0634 0402 0202          LI    R2,10             Get radix in reg for speed
     0404 000A
0635 0406 0220          AI    RO,-'0'           Convert from ASCII to binary
     0408 FFD0
0636 040A 8080          C     RO,R2             Is the character a digit?
0637 040C 1A08          JL    GETI02            Yes - there is a number!
0638 040E 04C0          CLR   RO                No - indicate no number
0639 0410 0453          B     *R3               Done -- no number
0640 0412
0641 0412 3802  GETI01  MPY   R2,RO             Multiply previous by radix
0642 0414 C000          MOV   RO,RO             Overflow?
0643 0416 16C4          JNE   ERRBLN            Yes - bad line number
0644 0418 C001          MOV   R1,RO             Get low order word of product
0645 041A A008          A     R8,RO             Add in next digit
0646 041C 11C1          JLT   ERRBLN            If number went negative-error
0647 041E 06A0  GETI02  BL    @GETCHR           Get next character
     0420 0254'
0648 0422 D501          MOVB  R1,*R4            Put into normal position
0649 0424 1306          JEQ   GETI03            If read end of line
0650 0426 0228          AI    R8,-'0'           Convert from ASCII to binary
     0428 FFD0
0651 042A 8088          C     R8,R2             Is this character a digit?
0652 042C 1AF2          JL    GETI01            Yes - try to pack it in
0653 042E 0620          DEC   @VARW             No point to 1st char after num
     0430 03B8'
0654 0432 04C8  GETI03  CLR   R8                Clean up our mess
0655 0434 C000          MOV   RO,RO             Hit a natural zero?
0656 0436 13B4          JEQ   ERRBLN            Yes-its an error
0657 0438 0453          B     *R3               And return
```

```
0659                    *                    THE LINE NUMBER TABLE
0660                    *       ALL TOKENS WHICH APPEAR IN THE TABLE MUST HAVE
0661                    *       NUMERICS WHICH FOLLOW THEM CRUNCHED AS LINE NUMBERS
0662 043A
0663 043A    81   LNTAB   BYTE ELSE$
0664 043B    85   GO$TOK  BYTE GO$
0665 043C    86           BYTE GOTO$
0666 043D    87           BYTE GOSUB$
0667 043E    88           BYTE RETUR$
0668 043F    8E           BYTE BREAK$
0669 0440    8F           BYTE UNBRK$
0670 0441    94           BYTE RESTO$
0671 0442    A5           BYTE ERROR$
0672 0443    A9           BYTE RUN$
0673 0444    B0           BYTE THEN$
0674 0445    ED           BYTE USING$
0675 0446    FF           BYTE >FF                Indicate end of table
0676 0448                 EVEN
```

```
0678                ***************************************************************
0679                *       Table of specially crunched statements
0680                *       2 bytes / special token                                 *
0681                *       Byte 1 - token value                                    *
0682                *       Byte 2 - "address" of special handler                   *
0683                *                Offset from label OFFSET in this assembly      *
0684                *                of the special case handler                    *
0685                ***************************************************************
0686 0448   02     SPECTB BYTE LIST$,CRU57-OFFSET
     0449   0C
0687 044A   05            BYTE OLD$,CRU58-OFFSET
     044B   28
0688 044C   07            BYTE SAVE$,CRU58-OFFSET
     044D   28
0689 044E   08            BYTE MERGE$,CRU58-OFFSET
     044F   28
0690 0450   82            BYTE SSEP$,CRU53-OFFSET
     0451   00
0691 0452   83            BYTE TREM$,CRU74-OFFSET
     0453   3C
0692 0454   93            BYTE DATA$,CRU58-OFFSET
     0455   28
0693 0456   9A            BYTE REM$,CRU74-OFFSET
     0457   3C
0694 0458   9D            BYTE CALL$,CRU66-OFFSET
     0459   6C
0695 045A   A1            BYTE SUB$,CRU65-OFFSET
     045B   5C
0696 045C   A3            BYTE IMAGE$,CRU54-OFFSET
     045D   22
0697 045E   FF            BYTE >FF
0698 0460                 EVEN
0699                      END
NO ERRORS,      NO WARNINGS
```

LABEL		VALUE	DEFN	REFERENCES						
$		0460'		0244	0351	0354				
ARG	R	035C'	0058	0507	0515	0516	0521	0523		
ARG2	R	02B4'	0059	0444	0452					
ARG4	R	0220'	0060	0159	0318	0385	0387			
BACKUP		03AC'	0566	0209	0256	0415	0429	0484	0492	0585
BCRU28		03A8'	0559	0473	0528	0545				
BERRSY		00EE'	0237	0328						
BREAK$		008E	0078	0668						
BUFLEV	R	002E'	0057	0158						
C4	R	01CC'	0059	0352						
CALL$		009D	0083	0694						
CBH20		01CB'	0351	0189						
CHAT	R	010E'	0057	0248						
COLON$		00B5	0091	0548						
COMMA		002C	0065	0434	0450	0459	0476			
COMMA$		00B3	0090	0339	0464					
CPALNM		0022	0109	0422	0488					
CPALPH		0020	0106	0109	0500					
CPBRK		0040	0107							
CPDIG		0002	0102	0109						
CPMO		0010	0105							
CPNIL		0000	0101							
CPNUM		0004	0103							
CPOP		0008	0104							
CPSEP		0080	0108							
CPTBL	R	00AE'	0062	0182	0210					
CRNADD		0102'	0244	0376						
CRNBUF	R	018C'	0057	0245	0327					
CRNEND	R	03CC'	0058	0590						
CRU01		002C'	0158	0150						
CRU02		004A'	0167	0163						
CRU04		0054'	0175	0355	0540					
CRU05		0058'	0176	0343						
CRU06		005C'	0177	0185	0217	0231	0380			
CRU08		0060'	0178	0199	0390					
CRU10		006A'	0182	0155	0169					
CRU100		0360'	0527	0344						
CRU105		0380'	0540	0531						
CRU106		0384'	0543	0367						
CRU12		006E'	0183	0181						
CRU14		0070'	0184	0321						
CRU15		0074'	0186	0192						
CRU16		008A'	0196	0167	0176	0388	0493			
CRU18		00BA'	0216	0203						
CRU20		00C0'	0220	0205						
CRU22		00D0'	0227	0368	0431					
CRU24		00E2'	0232	0222						
CRU26		00EC'	0235	0223	0233					
CRU27		00F2'	0239	0334	0549					
CRU28		00FA'	0242	0147	0166	0190	0197	0353	0366	0559
CRU32		0116'	0256	0202	0208	0211	0220	0401		
CRU32L		0244'	0401	0426						
CRU36		011A'	0257	0213						
CRU38		0124'	0261	0430						
CRU40		014A'	0301	0310						
CRU42		0160'	0308	0312						
CRU47		0190'	0329	0324	0332					
CRU48		01A2'	0336	0338						
CRU50		01B4'	0343	0340	0345					
CRU52		01B8'	0344	0337	0397					

CRU53	01D2'	0355	0342	0690							
CRU53A	01D6'	0357	0331								
CRU54	01F4'	0371	0696								
CRU57	01DE'	0363	0686								
CRU58	01FA'	0375	0687	0688	0689	0692					
CRU59	01FE'	0376	0372								
CRU60	020A'	0379	0466								
CRU61	0212'	0383	0264								
CRU61A	0218'	0385	0309								
CRU62	0222'	0388	0440								
CRU64	022A'	0390	0418	0438	0469	0497					
CRU65	022E'	0393	0695								
CRU66	023E'	0399	0694								
CRU68	0248'	0406	0229								
CRU70	0264'	0416	0412								
CRU72	0266'	0417	0494								
CRU74	020E'	0380	0400	0409	0414	0423	0453	0460	0462	0485	0489
			0691	0693							
CRU76	026C'	0422	0216								
CRU79	0280'	0430	0428								
CRU80	02DA'	0469	0154	0379							
CRU82	0284'	0431	0471								
CRU83	0288'	0434	0153	0375							
CRU83A	0298'	0444	0456								
CRU83B	028E'	0437	0371	0448	0461	0474					
CRU83C	02B2'	0452	0449								
CRU84	02BA'	0456	0152	0378							
CRU85	02CC'	0463	0451								
CRU85A	02D0'	0464	0435	0477							
CRU86	02F4'	0481	0503	0508							
CRU87	0304'	0485	0517	0524	0538						
CRU88	02F2'	0478	0475								
CRU90	0310'	0492	0518	0520							
CRU92	0314'	0493	0447	0458	0491	0510					
CRU93	031A'	0497	0399	0543							
CRU94	0306'	0488	0502								
CRU96	032C'	0506	0168	0175	0234	0389					
CRU98	0336'	0509	0506								
CRU99	0346'	0516	0514								
CRU99A	034E'	0519	0512								
CRUNCH D	0000'	0133	0055								
DATA$	0093	0080	0692								
DX10	0001	0003	0004	0135	0269	0568	0593				
ELSE$	0081	0071	0663								
ERRBLN	03A0'	0555	0643	0646	0656						
ERRCIP	01C2'	0349	0326								
ERRCOD R	03A6'	0059	0348	0349	0350	0352	0552	0555	0557		
ERRIVN	01BE'	0348	0225								
ERRLTL	0398'	0552	0591								
ERRNGT	01C6'	0350	0407								
ERRNTL	01CA'	0352	0384								
ERROR$	00A5	0086	0671								
ERRSYN	03A4'	0557	0237	0499	0501	0522	0547				
FAC R	0186'	0057	0142	0162	0325						
GETCHR R	0420'	0060	0178	0206	0257	0410	0647				
GETIO1	0412'	0641	0652								
GETIO2	041E'	0647	0637								
GETIO3	0432'	0654	0649								
GETINT	03FE'	0632	0161	0529							
GETNB R	02A0'	0060	0145	0164	0364	0445					

```
GETSTK  R  0112'   0058  0249
GETV1   R  0236'   0060  0188  0395
GO$        0085    0074  0664
GO$TOK     043B'   0664  0396
GOSUB$     0087    0076  0666
GOTO$      0086    0075  0665
GRMWA   R  0154'   0061  0295  0297  0301  0303
GROM    R          0060
IMAGE$     00A3    0085  0696
JNESY1     0296'   0441  0377  0386
JNESYN     0358'   0522  0441
KEYTAB  R  0138'   0058  0294
LBCPMO  R  00B0'   0062  0210
LENGTH     03E4'   0612  0417  0463
LIST$      0002    0067  0686
LLC        0020    0099
LN$        00C9    0094  0532
LNTAB      043A'   0663  0335
MAXKEY     000A    0097  0263
MERGE$     0008    0070  0323  0333  0689
OFFSET     01D2'   0354  0359  0686  0687  0688  0689  0690  0691  0692  0693
                         0694  0695  0696
OLD$       0005    0068  0687
P359       0000    0003  0003
PRGFLG  R  0036'   0058  0160
PUTCHR     03C6'   0589  0177  0212  0228  0239  0243  0259  0363  0482  0533
                         0536  0618
PUTSTK  R  0006'   0058  0141
QUOTE      0022    0064  0224  0365  0408  0413  0470
QUOTE$     00C7    0092  0227
R0         0000          0162  0179  0182  0198  0198  0201  0204  0221  0245
                         0246  0247  0302  0311  0322  0329  0330  0335  0336
                         0357  0385  0422  0437  0437  0488  0498  0498  0500
                         0511  0513  0515  0521  0530  0530  0534  0580  0581
                         0613  0614  0619  0633  0635  0636  0638  0641  0642
                         0642  0644  0645  0655  0655
R0LB    R  010C'   0059  0248
R1         0001          0146  0165  0180  0189  0207  0210  0258  0357  0358
                         0359  0365  0396  0411  0411  0413  0446  0446  0450
                         0589  0590  0602  0603  0644  0648
R10        000A          0153  0226  0367  0371  0375  0416  0448  0461  0474
R11        000B          0133  0612  0632
R12        000C          0133  0250
R13        000D          0138  0295  0297  0298  0299  0301  0303  0308  0314
R15        000F          0304  0306  0577  0579  0601  0603
R1LB    R  03D2'   0059  0601
R2         0002          0152  0265  0266  0267  0294  0295  0296  0297  0298
                         0301  0305  0307  0378  0478  0634  0636  0641  0651
R2LB    R  0152'   0059  0299  0303
R3         0003          0138  0149  0186  0187  0261  0262  0263  0265  0302
                         0305  0325  0383  0393  0394  0612  0620  0632  0639
                         0657
R4         0004          0143  0146  0165  0180  0196  0196  0258  0314  033C
                         0336  0427  0427  0605  0648
R5         0005          0230  0483  0615  0617
R6         0006          0168  0175  0234  0235  0344  0389
R6LB    R          0059
R7         0007          0154  0167  0176  0183  0216  0226  0229  0321  0379
                         0388  0399  0416  0478  0493  0502  0506
R8         0008          0144  0160  0182  0184  0184  0224  0227  0232  0242
```

				0323	0333	0339	0341	0406	0406	0408	0425	0434
				0439	0439	0457	0457	0459	0464	0470	0472	0472
				0476	0481	0490	0490	0509	0509	0519	0527	0527
				0532	0534	0535	0537	0544	0544	0546	0548	0614
				0615	0616	0633	0645	0650	0651	0654		
R8LB	R	000E'	0059	0143								
R9		0009		0200	0262	0306	0313	0327				
R9LB	R	0158'	0059	0304								
RAMPTR	R	03FA'	0057	0186	0191	0200	0230	0247	0261	0313	0376	0393
				0483	0553	0589	0604	0613	0617	0619		
REM$		009A	0082	0693								
RESTO$		0094	0081	0670								
RETUR$		0088	0077	0667								
RUN$		00A9	0087	0672								
SAVE$		0007	0069	0688								
SPECTB		0448'	0686	0322								
SSEP$		0082	0072	0690								
SUB$		00A1	0084	0695								
THEN$		00B0	0088	0673								
TO$		00B1	0089	0341								
TREM$		0083	0073	0691								
UNBRK$		008F	0079	0669								
UNQST$		00C8	0093	0481								
USING$		00ED	0095	0674								
VAR5	R	02D6'	0058	0465								
VARA	R		0057									
VARW	R	0430'	0057	0240	0444	0452	0566	0579	0653			
VARW1	R	03B2'	0057	0577								
VDPRD	R	03C0'	0061	0308	0581							
VDPWD	R	03E0'	0061	0605								
VERMAC	M		A0001	0003								
VERS		0000	0003	0004	0135	0269	0568	0593				
VRAM	R		0060									
WRVDP	R	03D6'	0061	0602								