```
1   *    GENERAL AUTOMATION, INC.  ALL RIGHTS RESERVED
2   ***************************************************
3   *
4   *    PROGRAM NAME  FPH-09
5   *
6   *    MODEL NUMBER  8F009
7   *
8   *    PURPOSE       FORTRAN PHASE-09
9   *
10  *    PROGRAMMER    DICK WALLMANN, MODS-MARK ELFIELD
11  *
12  ****************   REVISION LIST   **************
13  *
14  *    RV DATE     SCO   BY  REASON FOR CHANGE
15  *    -- -------- ----- --- ----------------------
16  *
17  *    01 11/16/70 NONE  RPH INITIAL RELEASE
18  *
19  ***************************************************
20  ***************************************************
21        HDNG    MPX FORTRAN ** DATA STATEMENT
22  ***************************************************
23  *STATUS-VERSION 1, MODIFICATION 0
24  *
25  *FUNCTION/OPERATION-
26  *    * EXAMINES ONLY DATA STATEMENTS, BYPASSING ALL
27  *      OTHER STATEMENTS,
28  *    * CHECKS EACH VARIABLE FOR VALIDITY
29  *    * CHECKS TO SEE THAT EACH VARIABLE NAME HAS
30  *      BEEN ENTERED INTO THE SYMBOL TABLE,
31  *    * CHECKS TO ENSURE THAT DIMENSIONING INDICATED
32  *      IN THE DATA STATEMENT DOES NOT EXCEED THE
33  *      DIMENSIONS INDICATED BY THE SYMBOL TBL ENTRY
34  *    * CHECKS EACH DATA STMNT FOR CORRECT SYNTAX
35  *    * REFORMATS THE DATA STMNT INTO A STRING OF
36  *      DATA GROUPS,
37  *
38  *ENTRY POINTS-
39  *    * BEGIN-PHASE 9 IS READ INTO CORE BY PHASE 8
40  *            VIA THE ROLRX ROUTINE,  EXECUTION IS
41  *            BEGUN AT LOCATION BEGIN,
42  *
43  *INPUT-
44  *    * THE STATEMENT STRING
45  *    * THE SYMBOL TABLE
46  *
47  *OUTPUT-
48  *    * THE UPDATED STATEMENT STRING
49  *    * THE UPDATED SYMBOL TABLE
50  *
51  *EXTERNAL REFERENCES-
52  *    SUBROUTINES-
53  *      * ROLRX
54  *    OTHER FORTRAN PHASES-
55  *      * NONE
56  *
57  *EXITS-
58  *    NORMAL-
59  *      * EXITS TO THE SUCCEEDING PHASE VIA A CALL
```

```
60  *        TO THE ROLRX ROUTINE.
61  *
62  *    ERROR-
63  *       * OVERLAP-DOES NO PROCESSING AND EXITS
64  *         NORMALLY
65  *       * ERRORS DETECTED BY THIS PHASE ARE
66  *         75, 76, 77, 78, 79, 80, AND 82.
67  *
68  *TABLES/WORK AREAS-
69  *       * THE STATEMENT STRING, THE SYMBOL TABLE,
70  *         AND THE FORTRAN COMMUNICATIONS AREA.
71  *
72  *ATTRIBUTES-NONE
73  *
74  *NOTES-N/A
75  *
76  ************************************************************
77         HDNG    MPX FORTRAN ** DATA STATEMENT
78  *
79         ABS     REF   CORE
80  *
81  *      SYSTEM AND FORTRAN EQUATES
82  *
83  MEMRY  EQU     ~~17FFF~~ CORE      MAXIMUM CORE SIZE
84  PHSIZ  EQU     4*320               MAXIMUM PHASE SIZE
85  OVERL  EQU     MEMRY-PHSIZ         PHASES 2-29 START
86  FCOM   EQU     OVERL-22            FORTRAN COMM. TABLE
87  PHNTB  EQU     FCOM-56             PHASE TABLE
88  ROLRX  EQU     PHNTB-50            INTERPHASE CALL
89  *
90  *      FORTRAN COMMUNICATION AREA
91  *
92         ORG     FCOM        BEGIN PHASE
93  *
94  SOFS   BSS     1           START OF STRING
95  EOFS   BSS     1           END OF STRING
96  SOFST  BSS     1           START OF SYMBOL TABLE
97  SOFNS  BSS     1           START OF NON-STMNT NUMBERS
98  SOFXT  BSS     1           START OF SUBSC TEMPS
99  SOFGT  BSS     1           START OF GENERATED TEMPS
100 EOFST  BSS     1           END OF SYMBOL TABLE
101 COMON  BSS     1           NEXT AVAILABLE COMMON
102 CSIZE  BSS     1           SIZE OF COMMON
103 ERROR  BSS     1           OVERLAP ERROR
104 FNAME  BSS     1           PROGRAM NAME
105        BSS     1           *
106 SORF   BSS     1           SUBR (-) OR FUNCTION (+)
107 CCWD   BSS     1           CONTROL CARD WORD
108 *                  BIT 15 TRANSFER TRACE
109 *                  BIT 14 ARITHMETIC TRACE
110 *                  BIT 13 EXTENDED PRECISION
111 *                  BIT 12 LIST SYMBOL TABLE
112 *                  BIT 11 LIST SUBPROGRAM NAMES
113 *                  BIT 10 LIST SOURCE PROGRAM
114 *                  BIT  9 ONE WORD INTEGERS
115 *                  BIT  8 PUNCH
116 *                  BIT  7 NONPROCESS PROGRAM
117 IOCS   BSS     1           IOCS CONTROL CARD WORD
118 *
119 *        SEE PHASE ONE FOR BIT PATTERNS
```

```
120   *
121   DFCNT BSS     1          DEFINE FILE COUNT
122   *
123   LCOMN BSS     2          SIZE OF INSKEL COMMON
124   *
125   ICCER BSS     2          IOCS CONTROL CARD ERROR
126   *
127         BSS     2          SYSTEM LOADER USE
128   *
129   *                    END OF FORTRAN COMMUNICATION
130   *                    AREA
131         HDNG    MPX FORTRAN ** DATA STATEMENT
132         ORG     OVERL      PHASE ORIGIN
133   *
134   *
135   * THIS PHASE PROCESSES THE DATA STATEMENT, INPUT I
136   *
137   *     ST-ID NAME1,,,,,NAMEN/CON1,,,,CONN/
138   * WHERE NAMES HAVE NOT YET BEEN LOOKED UP IN THE
139   * SYMBOL LIST, REAL CONSTANTS ARE ALREADY CONVERTE
140   * EXCEPT FOR SIGN, AND INTEGERS ARE NOT YET
141   * CONVERTED, OUTPUT IS OF THE FORM
142   *
143   *     ST-ID DATAGROUP HDR CON1 NAME1 DATAGROUP HDR
144   *     CON2 NAME2,,,DATAGROUP HDR CONN NAMEN
145   *
146   * WHERE THE DATAGROUP HDR IS OF THE FORM BIT0 0,
147   * BIT 1-7 DUPLICATION FACTOR, BITS 8-15 CONSTANT
148   * LENGTH, BIT 1 OF THE NAME PT 1 IF A DISPLACEMENT
149   * WORD FOLLOWS, ELSE IT IS 0, ALL NAME PTS NOW
150   * POINT TO THE SYMBOL LIST, AND ALL CONSTANTS ARE
151   * CONVERTED,
152         HDNG    MPX FORTRAN ** DATA STATEMENT
153   BEGIN LD      ERROR      OVERLAP ERROR SWITCH
154         BSC   L EXIT1,Z    SKIP PHASE IF OVERLAP ERRO
155         LD      SOFS       START OF STRING ADDRESS
156         STO   L NEXT       NEXT STMNT ID WORD ADDRESS
157         LDX   L1 WORK      INDEX REG 1  WORK AREA ADD
158         BSI     FINDD      FIND FIRST DATA STMNT
159         LD      START      CURRENT STMNT ID WD ADDR
160         STO   L ENDD       PRESET FOR MOVE
161         LD      CCWD       CONTROL CARD WORD
162         AND     L4         EXTRACT BIT 13= PRECISION
163         SRA     2          RIGHT JUSTIFY
164         A       TWO        LENGTH 2 NORMAL, 3 EXTENDE
165         STO   L LNGTH      REAL WORD LENGTH
166         STO     INTL       SAVE AS INTEGER LENGTH
167         LD      CCWD       CONTROL CARD WORD
168         SLA     9          ONE=WD INTEGER INDR TO SIG
169         SRA     15         RIGHT JUSTIFY INDICATOR
170         BSC     Z          SKIP IF NO ONE=WD INTEGERS
171         STO     INTL       SET INTEGER LENGTH TO ONE
172   *
173   *               FIND FIRST CONSTANT IN SUBSTATEMENT
174   *
175   FINDC STX   L2 K           SET K J
176         STX    2 J         SAVE J
177   FNDC1 EQU     *           LABEL FNDC1 EQUALS LOOP1
178   LOOP1 LD    I K           NEXT STMNT WORD
179         S       SLASH      SLASH CONSTANT
```

```
180          BSC  L  LOOP2, -   BRANCH IF SLASH
181          MDX  L  K,1        INCREMENT K K 1
182          LD      NEXT       ADDR NEXT STMNT ID WORD
183          S    L  K          ADDR CURRENT STMNT POINTER
184          BSC  L  LOOP1,Z    BR IF NOT STMNT END
185  ERR8    LDX  X1 D8         ILLEGAL STATEMENT FORMAT
186          BSC  L  ERR        BR TO SET UP ERROR 75
187  *
188  *               CONSTANTS AND WORK AREA
189  *
190  J       DC       *-*       INDEX FOR VARIABLES
191  HDRPT   DC       *-*       DATA GROUP HEADER WD PT
192  L4      DC       4         DECIMAL 4
193  *
194  *               CHECK FOR NAMES
195  *
196  LOOP2   STO  L  SUB        CLEAR SUBSCRIPT
197          LD   L  K          CURRENT POINTER ADDRESS
198          S       J          POINTER AFTER LAST CONSTAN
199          SRA     1
200          BSC  L  NEXTC 1,Z  BR IF NAMES SPECIFIED
201          LDX  X1 D1         NO NAMES SPECIFIED
202          BSC  L  ERR        BR TO SET UP ERROR 75
203  NEXTC   EQU      *         LABEL FOR NEXT INSTRUCTION
204          STX   2 J          SAVE POINTER AFTER LAST CO
205          MDX  L  K,1        INCR INTER-STMNT POINTER
206          STX   1 HDRPT      HEADER WORD POINTER
207          BSI     INCRP      INCR WORK AREA POINTER
208          BSI  L  GCON       BR TO GET NEXT CONSTANT
209          LD      CONL       CONSTANT LENGTH INDR
210          BSC  L  ERR8, -    BR IF CONSTANT NOT FOUND
211          LDX   2 1
212          STX   2 DUP        DUPLICATION FACTOR 1
213          LD   I  K          NEXT STMNT WORD
214          S       STAR       ASTERISK CONSTANT
215          BSC  L  BLDH,Z     BR IF NOT ASTERISK
216          LD      CONL       CONSTANT LENGTH INDICATOR
217          S    L  ONE
218          BSC  L  * 3, -     BR IF CONSTANT LENGTH 1
219  ERR2    LDX  X1 D2         NOT A POSITIVE INTEGER
220          BSC  L  ERR        BR TO SET UP ERROR 75
221          LD    1 -1         LAST WORD HEADER WORK AREA
222          BSC  L  ERR2,Z     BR TO SET ERROR 75
223          STO     DUP        SAVE FOR NAME ITERATION
224          S       TOP        CHECK DUPLICATION SIZE
225          BSC  L  ERR2,-     BR IF TOO LARGE
226          MDX  L  K,1        INCREMENT K
227          MDX   1 -1         MOVE WORK AREA POINTER
228          BSI  L  GCON       GET NEXT CONSTANT
229          LD      CONL       CONSTANT LENGTH INDR
230          BSC  L  BLDH,Z     BR IF CONSTANT FOUND
231          LDX  X1 D3         SET UP ERROR 75 IF CONSTAN
232          BSC  L  ERR        NOT FOUND AFTER DUP FACTOR
233  *
234  *               CONSTANTS
235  *
236  STAR    DC       /1C       ASTERISK CONSTANT
237  TOP     DC       /1000     DUPLICATION FACTOR LIMIT
238  *
239  * THIS ROUTINE INCREMENTS X1, THE WORK AREA POINTE
```

```
240   * BY ONE, CHECKING TO SEE THAT THE LIMIT OF THE
241   * AREA IS NOT EXCEEDED, IF IT IS EXCEEDED, THE
242   * STATEMENT RECEIVES AN ERROR 82.
243   *
244   INCRP DC      *-*       RETURN ADDR
245         MDX   1 1         INCR WORK AREA POINTER
246         STX   1 WKAA
247         LD      WKAA      WORK AREA POINTER
248         S       WKNDA     WORK AREA END ADDRESS
249         BSC   I INCRP, Z  EXIT IF NO OVERLAP
250         LDX   1 D15       OVERLAP ERROR INDICATED
251         BSC   L ERR       SET UP ERROR 82
252   *
253   *             CONSTANT AND WORK AREA
254   *
255   WKNDA DC      WKEND     ADDR END OF WORK AREA
256   WKAA  DC      *-*       CURRENT END OF WORK AREA
257   *
258   * THIS ROUTINE LOCATES THE NEXT DATA STATEMENT,
259   * THE POINTERS START AND NEXT ARE UPDATED TO POINT
260   * TO THE START OF THE NEXT DATA STATEMENT AND TO
261   * THE STATEMENT FOLLOWING, RESPECTIVELY.
262   *
263   FINDD DC      *-*       RETURN ADDRESS
264         LDX   I2 NEXT     POINTER TO NEXT STMNT ID W
265         STX   L2 SAVE 1   SAVE FOR MOVE
266   FNDD1 STX    2 START    SAVE AS START OF STMNT
267   START EQU     * 1       LABEL NEXT INSTRUCTION ADD
268         LD    L *-*       STMNT ID WORD, MODIFIABLE
269         SRA     2         RIGHT JUSTIFY STMNT WD CNT
270         AND     L1FF      EXRACT WORD COUNT
271         A       START     START OF STMNT ADDR
272         STO     NEXT      ADDR OF NEXT STMNT ID WORD
273         S     L EOFS      END OF STRING ADDRESS
274         BSC   L EXIT,Z-   EXIT IF END OF STRING
275         LD     2 0        STMNT ID WORD
276         SRA     11        RIGHT JUSTIFY STMNT TYPE
277         S       DATA      DATA STMNT TYPE CONSTANT
278         BSC   I FINDD, -  EXIT IF DATA STMNT
279   NEXT  EQU     * 1       LABEL NEXT INSTRUCTION ADD
280         LDX   L2 *-*      SET PT STMNT ID, MODIFIABL
281         MDX     FNDD1     BR TO FIND NEXT STMNT ID W
282   *
283   *             CONSTANTS AND WORK AREA
284   *
285   L1FF  DC      /1FF      STMNT WD COUNT MASK
286   DATA  DC      /1F       DATA STMNT TYPE CODE
287   H8000 DC      /8001     NAME BIT PLUS 1
288   H20   DC      /20       DEFINED BIT
289   TWO   DC      2         DECIMAL TWO
290   CONL  DC      *-*       LENGTH OF CONSTANT
291   INTL  DC      *-*       INTEGER ELEMENT LENGTH
292   SLASH DC      /21       / CONSTANT
293   *
294   *             BUILD DATAGROUP HEADER WORD
295   *
296   BLDH  LD      CONL      CONSTANT LENGTH
297         STO     CONLT     SAVE FOR NAME ITERATION
298         LD      DUP       DUPLICATION FACTOR
299         SLA     3         TO HIGH 13 BITS
```

```
300          A        CONL        CONSTANT LENGTH
301          STO   I  HDRPT       DATAGROUP HEADER WORD
302          LD       CONTP       CONSTANT TYPE
303          STO      CONTT       SAVE FOR NAME ITERATION
304          LDX   I2 J           SET INDEX REG 2   J
305          MDX   L  SUB,0       CHECK SUBSC TEMP=SKIP IF 0
306          MDX      NXTN4 1     BR IF NAME NOT YET FULL
307 NEXTN    MDX   2  1           INCR J J 1
308          LD    2  0           NEXT STMNT WORD
309          BSC   L  ERR2,-      BR IF NOT A NAME      *V1M3
310          LDX   I3 SOFNS       INDEX REG 3 SYMBOL TABLE P
311 NXTN1    LD    2  0           NEXT STMNT WORD
312          S     3  1           SYMBOL TABLE ENTRY
313          BSC   L  NXTN3, -    BR IF NAME FOUND
314 NXTNA    EQU      *           LABEL NEXT INSTRUCTION
315          LD    3  0           EXTRACT DIMENSION
316          SLA      3           *INFORMATION
317          SRA      14          *
318          BSC      Z           SKIP NEXT IF NO DIMENSION
319          MDX   3  -3          DECR SYMBOL TABLE POINTER
320          MDX   3  -3          DECR SYMBOL TABLE POINTER
321          STX   3  NOST        ADDR TEMP
322          LD       NOST        CURRENT SYMBOL TBL POINTER
323          S     L  EOFST       END OF SYMBOL TABLE ADDRES
324          BSC   L  NXTN1,Z     BR IF NOT STRING END
325 ERR4     LDX   X1 D4          NAME IS UNDEFINED
326          BSC   L  ERR         SET UP ERROR 80       *V1M3
327 *
328 *                 CONSTANTS AND WORK AREA
329 *
330 THREE    DC       3           DECIMAL 3 CONSTANT
331 DUP      DC       *-*         DUPLICATION FACTOR
332 NOST     DC       *-*         INTEGER SAVE TEMPORARY
333 CONLT    DC       *-*         CONSTANT LENGTH TEMPORARY
334 CONTT    DC       *-*         CONSTANT TYPE TEMPORARY
335 BLANK    DC       /4040       WORD OF EBCDIC BLANKS
336 CONTP    DC       *-*         CONSTANT TYPE
337 T8000    DC       /8000       NAME BIT CONSTANT
338 TNAME    DC       *-*         SAVE NAME TEMPORARILY HERE
339 *
340 *                 CONTINUE BUILD DATAGROUP HEADER WORD
341 *
342 NXTN3    LD    2  1           NEXT STMNT WORD
343          BSC      -           SKIP IF 2ND WORD OF NAME
344          LD       T8000       BLANK 2ND WORD OF NAME
345          S     3  2           2ND WD FR SYMBOL TABLE
346          BSC   L  NXTNA,Z     BR IF NAME NOT FOUND
347          LD    2  1           2ND HALF OF NAME
348          BSC      Z           SKIP NEXT IF NOT NAME
349          MDX   2  1           INCR STMNT POINTER J J 1
350 NXTN4    EQU      *           LABEL FOR NEXT INSTRUCTION
351          MDX   2  1           INCR STMNT POINTER
352          STX   3  NOST        SYMBOL TABLE POINTER
353          LD    L  SOFST       START OF SYMBOL TABLE ADDR
354          S        NOST        CURRENT POINTER VALUE
355          SRT      16          MAKE NORMAL DIVIDEND
356          D        THREE       NO. OF WORDS PER ENTRY
357          A        H8000       NAME BIT
358          STO      TNAME       SAVE NAME
359          LD    3  0           SYMBOL TABLE ID WORD
```

```
360          OR       H20          SET DEFINED BIT
361          STO   3  0            RESTORE ID WORD
362          BSC   L  ERR9,       BR IF NOT A VARIABLE- ERRO
363          SRA      14           DATA TYPE BITS
364          STO      NOST
365          S        CONTT        CON TYPE TEMPORARY
366          BSC   L  NXTN8, -     BR IF SAME TYPE
367          LD       CONTT        CONSTANT TYPE TEMP
368          S        TWO
369          BSC   L  * 3, -       BR IF HOLLERITH DATA
370  ERR9    LDX   1  D9           DATA TYPES DO NOT MATCH
371          BSC   L  ERR          SET UP ERROR 77
372  NXTN8   LD    L  LNGTH        REAL VARIABLE LENGTH
373          MDX   L  NOST,0       SKIP IF REAL VARIABLE
374          LD       INTL         INTEGER LENGTH
375          STO      NAML         SAVE AS ELEMENT LENGTH
376          S        CONLT        CONSTANT LENGTH TEMPORARY
377          BSC   L  ERR11, Z     BR TO SET UP ERR 77 IF NEG
378          BSC   L  NXT8A,+      BR IF ZERO (NEG BR ABOVE)
379          LD       CONTT        GET CONSTANT TYPE TEMP
380          S        TWO          COMPARE TO TWO
381          BSC   L  NXT8A,Z      BR TO STORE NAME IF NO COM
382          LD       BLANK        ELSE GET A BLANK
383          STO   1  0            PUT THE BLANK IN STRING
384          BSI   L  INCRP        GO MOVE POINTER
385          LD    I  HDRPT        GET DATA GROUP HE ADER WOR
386          A     L  ONE          INCREASE IT BY ONE
387          STO   I  HDRPT        REPLACE THE MODIFIED WORD
388          MDX   L  CONLT,1      INCREASE CONSTANT LENGTH
389          MDX      NXTN8        LOOP BACK
390  NXT8A   LD       TNAME        GET THE NAME SAVED EARILER
391          STO   1  0            PUT NAME ON STRING
392          BSI   L  INCRP        GO MOVE POINTER
393          LD    3  0            SYMBOL TABLE ID WORD
394          SLA      3            EXTRACT AND RIGHT JUSTIFY
395          SRA      14           DIMENSION BITS,
396          STO      DIMN         SAVE DIMENSIONALITY
397          STO      DIMNT        SAVE FOR SUBSC CALCULATION
398          BSC   L  NXTN7, -     BR IF NOT DIMENSIONED
399          BSI   L  INCRP        INCR WORK AREA POINTER
400          MDX   L  SUB,0        TEST FOR SUBSCRIPT
401          MDX      NXTN2        BR IF SUBSCRIPT
402          MDX   1  -1           DECR WORK AREA POINTER
403          LD    2  0            NEXT STMNT WORD
404          S        LPAR         LEFT PARENTHESIS CONSTANT
405          BSC   L  NXTN5, -     BR TO GET SUBSCRIPT
406  NXTN7   LD    L  ONE
407          MDX   L  DIMN,0       SKIP IF NOT DIMENSIONED
408          LD    3  -3           NEXT SYMBOL ID WORD
409          S        SUB          REMAINING ELEMENTS IN NAME
410          BSC   L  * 3,-Z       BR IF SUBSCRIPT OK
411  ERR13   EQU      *            LABEL FOR NEXT INSTRUCTION
412          LDX   X1 D13          SUBSCRIPT TOO LARGE
413  GERR    BSC   L  ERR          SET UP ERROR 33
414          S        DUP          DUPLICATION FACTOR
415          STO      TEMP         REMAINING AFTER THIS CON
416          BSC   L  NXTN9,       BR IF NAME FULL OR OVERFLO
417          LD    3  -3           NEXT ENTRY ID WORD
418          S        TEMP         REMAINDER
419          STO      SUB          SUBSCRIPT FOR NEXT CONSTAN
```

```
420          MDX      COM          BR TO TEST FOR ANOTHER CON
421  NXTN9   SRA      16
422          STO      SUB          CLEAR SUBSCRIPT
423          S        TEMP         REMAINDER
424          STO      DUP          DUPCTN FACTOR FOR NEXT NAM
425          BSC  L   * 5, -       BR IF CONSTANT EXHAUSTED
426          LD     2 0            NEXT STMNT WORD
427          S        COMMA        COMMA CONSTANT
428          BSC  L   NEXTN, -     BR CONSTANT NOT EXHAUSTED
429          MDX      ERR7         NO COMMA AFTER NAME
430          LD     2 0            NEXT STMNT WORD
431          S        COMMA        COMMA CONSTANT
432          BSC  L   COM, -       BR IF COMMA
433  SLSH2   EQU      *            LABEL FOR NEXT INSTRUCITON
434          LD     2 0            NEXT STMNT WORD
435          S    L   SLASH        SLASH CONSTANT
436          BSC  L   SLSH, -      BR IF SLASH
437          LDX    X1 D5          ILLEGAL CHAR AFTER NAME
438          MDX      GERR         BR TO SET UP ERROR 75
439  *
440  *                CONSTANTS AND WORK AREA
441  *
442  SUB     DC       0            SUBSCRIPT TEMPORARY
443  TEMP    DC       *-*          REMAINDER TEMPORARY
444  LPAR    DC       /0D          LEFT PARENTHESIS CONSTANT
445  H4000   DC       /4000        NAME SUBSCRIPT BIT
446  NAML    DC       *-*          CURRENT DIMENSION LENGTH
447  DIMNT   DC       0            DIMENSION TEMPORARY
448  COMMA   DC       /2B          COMMA CONSTANT
449  SUBSW   DC       *            SUBSCRIPTED ELEMENT SWITCH
450  *
451  *                CALCULATE VECTOR DISPLACEMENT FROM
452  *                VARIABLE STATEMENT
453  *
454  NEXTS   LD     2 0            NEXT STMNT WORD
455          S        COMMA        COMMA CONSTANT
456          BSC  L   ERR13,Z      BR NO COMMA BETWEEN SUBSC
457          MDX    1 -1           DECR WORK AREA POINTER
458          BSI  L   GCONS        GET NEXT SUBSCRIPT INTEGER
459          M      3 0            MPY BY DIMENSION
460          RTE      16           PRODUCT TO ACC
461          A        SUB          ACCUMULATED VALUE
462          MDX      TESTS        BR TO SAVE AND TEST SUBSC
463  NXTN5   STO      SUBSW        SET SUBSCRIPTED ELT SWITCH
464          LD   L   K
465          STO      TEMP         SAVE K
466          STX  L2  K            K J
467          BSI  L   GCONS        GET SUBSCRIPT INTEGER
468  TESTS   STO      SUB          SAVE ACCUMULATED RESULT
469          S      3 -1           TEST FOR CURRENT SIZE
470          BSC  L   ERR13,-      TOO LARGE
471          MDX    3 -1           MOVE DIMENSION PT BACK ONE
472          MDX  L   DIMNT,-1     TEST IF MORE SUBSC EXPECTE
473          MDX      NEXTS        YES
474  NXTN6   LD       TEMP         K FROM TEMPORARY
475          STO  L   K            RESET K
476          MDX    2 1            INCREMENT STMNT POINTER
477          LD     2 -1           LAST STMNT WORD
478          S    L   RPAR         RIGHT PARENTHESIS CONSTANT
479          BSC  L   ERR13,Z      BR NOT RIGHT PARENTHESIS
```

```
480  DIMN   EQU      * 1          NUMBER OF DIMENSIONS
481         MDX   L3 *-*          SYMBOL TBL PT, MODIFIABLE
482  NXTN2  LD       SUB          SUBSCRIPT TEMP
483         M        NAML         GET TRUE DISPLACEMENT
484         SLT      16           PRODUCT TO ACC
485         STO    1 -1           STORE SUBSCRIPT IN WORK
486         LD     1 -2           NAME WORD
487         OR       H4000        SET SUBSCRIPT BIT IN NAME
488         STO    1 -2           RESTORE NAME WORD
489         MDX   L  SUBSW,0      SKIP IF SUBSCRIPTED ELT
490         MDX      NXTN7        BR TO FIND MORE ENTRIES
491         STX      SUBSW        RESET SUBSCRIPTED ELT SW
492         SLA      16           LOAD ZERO
493         STO      DIMN         ZERO NO. OF DIMENSIONS
494         STO      DIMNT        ZERO WORK AREA
495         STO      SUB          ZERO SUBSCRIPTS
496         LD     1 -2           SET SUBSCRIPTED ELEMENT
497         OR       H2000        *BIT INTO NAME POINTER
498         STO    1 -2           *
499         MDX      NXTN7        BR TO FIND MORE
500  H2000  DC       /2000        SUBSCRIPTED ELEMENT BIT
501  COM    EQU      *            LABEL FOR NEXT INSTRUCTION
502         LD     I K            NEXT STMNT WORD
503         S        COMMA        COMMA CONSTANT
504         BSC   L  NEXTC, -     NEXT CON IF MORE IN LIST
505         MDX   L  SUB,0        SKIP NEXT SUBSCRIPT   ZERO
506         MDX      SLSH2        CHK FOR UNFILLED LAST NAME
507  ERR7   LDX   X1 D7           NO. OF CONS, NAMES UNEQUAL
508         MDX      ERR          BR TO SET UP ERROR 76
509  *
510  *               MOVE WORK STRING TO STRING AREA
511  *
512  SLSH   EQU      *            LABEL FOR NEXT INSTRUCITON
513         STO      SUB          CLEAR SUBSCRIPT
514         LD     I K            NEXT STMNT WORD
515         S     L  SLASH        SLASH CONSTANT
516         BSC   L  ERR7,Z       BR IF NOT SLASH
517         MDX   L  K,1          INCR STMNT POINTER
518         LD     I K            NEXT STMNT WORD
519         S        COMMA        COMMA CONSTANT
520         BSC   L  SLSH1,Z      BR IF NOT A COMMA
521         LD       K            STMNT POINTER
522         STO   L  J            J K
523         BSC   L  FNDC1        BR FOR MULTIPLE STMNTS
524  SLSH1  MDX   L  K,1          INCR STMNT POINTER
525         LD       K            CURRENT POINTER VALUE
526         S     L  NEXT         ADDR NEXT STMNT ID WORD
527         BSC   L  ERR8,Z       BR IF NOT STMNT END
528         MDX   L1 1-WORK       GET LENGTH
529         STX    1 SL 1         END OF OUTPUT INCREMENT
530         LDX   I2 ENDD         PICK UP END OF OUTPUT PT
531         LD       DATAS        DATA STMNT ID SHIFTED 2 RT
532         OR       SL 1         BUILD NEW STMNT ID FOR DAT
533         SLA      2            RESTORE TO NORMAL FROM
534         STO      IDWD         SAVE NEW ID WORD
535         S      2 0            END OF OUTPUT WORD
536         BSC   L  OPEN,-Z      BR TO OPEN STRING
537  CONTS  LD       IDWD         RESTORE NEW ID WORD
538         STO    2 0            MOVE IN DATA STMNT ID
539         MDX    1 -1           DECR WORK AREA POINTER
```

```
540  SL     MDX   L2 *-*        PT TO END O/P, MODIFIABLE
541         STX    2 ENDD       SAVE AS NEW END
542  LOOP3  LD    L1 WORK-1     WORD FROM WORK AREA
543         STO    2 -1         TO STRING AREA
544         MDX    2 -1         DECR STRING POINTER
545         MDX    1 -1         DECR AREA POINTER
546         MDX      LOOP3      BR IF WORK PT NOT ZERO
547  NXTD   BSI   L  FINDD      FIND NEXT DATA STMNT
548         BSI      MOVE       COMPRESS STRING
549         LDX   L1 WORK       RESET WORK POINTER
550         BSC   L  FINDC      FIND CONSTANT
551  *
552  *                WORK AREA
553  *
554  IDWD   DC       *-*        NEW ID WORD TEMPORARY
555  RUNOF  DC       *-*        STRING SIZE TO MOVE
556  *
557  *                THIS ROUTINE OPENS THE STRING TO PUT
558  *                ENLARGED DATA STMNT IN STRING AREA
559  *
560  OPEN   SRA      2
561         STO      CNTOF      COUNT OVERFLOW
562         A     L  EOFS       FIND NEW END OF STRING
563         STO      OWD1 3     PUT ADDR IN MOVER LOOP
564         LD    L  EOFS       END OF STRING ADDRESS
565         STO      OWD1 1     MOVE INSTRUCTION
566         S        ENDD       START OF MOVE AREA
567         A        ONE        ONE
568         STO      RUNOF      STRING SIZE TO MOVE
569         LD       OWD1 3     MODIFY ADDR IN MOVER LOOP
570         STO   L  EOFS       NEW END OF STRING ADDRESS
571  OWD1   LD    L  *-*        NXT WD TO MOVE, MODIFIABLE
572         STO   L  *-*        OPEN STRING, MODIFIABLE
573         MDX   L  OWD1 1,-1  DECR NEXT WORD TO MOVE ADD
574         MDX   L  OWD1 3,-1  DECR MOVE TO ADDRESS
575         MDX   L  RUNOF,-1   DECR NO, OF WORDS TO MOVE
576         MDX      OWD1       BR TO CONTINUE
577         LD    L  NEXT       ADDR NEXT STMNT ID WORD
578         A        CNTOF      RANGE OF MOVE
579         STO   L  NEXT       NEW NEXT STMNT ADDR
580         MDX      CONTS      BR TO RESTORE NEW ID WORD
581  *
582  CNTOF  DC       *-*        RANGE OF STRING OPENING
583  *
584  * THIS ROUTINE IS ENTERED FOR ALL ERROR CONDITIONS
585  * THE STATEMENT BEING PROCESSED IS DISCARDED, AND
586  * REPLACED BY AN ERROR RECORD, MOVE IS THEN CALLED
587  * TO CONDENSE THE STRING, AND PROCESSING CONTINUES
588  * WITH THE NEXT STATEMENT. ON INPUT, INDEX 1 HOLDS
589  * THE ERROR NUMBER.
590  *
591  ERR    LD       ERRCD      ERROR INDICATOR
592         STO   I  ENDD       SET ERROR IN STMNT ID WORD
593         MDX   L  ENDD,1     INCR END OF STRING
594         STX   I1 ENDD       SET ERROR CODE
595         MDX   L  ENDD,1     INCR END OF STRING
596         MDX      NXTD       COMPRESS, THEN NEXT STMNT
597  *
598  *                CONSTANTS AND ERROR EQUIVALENCE TABL
599  *
```

```
600  ERRCD DC       /A008      ERROR INDICATOR + STMNT ID
601  D1    EQU      75         NO NAMES SPECIFIED
602  D2    EQU      75         DUPCTN FACTOR NOT POS DIGI
603  D3    EQU      75         NO CON AFTER DUPCTN FACTOR
604  D4    EQU      80         UNDEFINED NAME
605  D5    EQU      75         ILLEGAL CHAR AFTER NAME
606  D6    EQU      24         ILLEGAL INTEGER CONSTANT
607  D7    EQU      76         NAMES AND CONS NOT 1 TO 1
608  D8    EQU      75         ILLEGAL STMNT FORMAT
609  D9    EQU      77         NAME AND CON TYPE NOT SAME
610  D10   EQU      75         ILLEGAL MINUS SIGN
611  D11   EQU      78         ILLEGAL HOLLERITH CONSTANT
612  D12   EQU      79         ILLEGAL PARTIAL WORD FIELD
613  D13   EQU      33         ILLEGAL SUBSCRIPT
614  D14   EQU      79         ILLEGAL CHAR IN PARTIAL WD
615  D15   EQU      82         WORK AREA EXCEEDED
616  K     DC       *-*        INDEX FOR CONSTANTS
617  DATAS DC       /3E00      DATA STMNT ID SHIFTED 2 RT
618  RPAR  DC       /1D        RIGHT PARENTHESIS CONSTANT
619  MINUS DC       /20        -
620  BCD   DC       /3D        QUOTE MARK  APOSTROPHE
621  ONE   DC       1          DECIMAL ONE CONSTANT
622  *
623  * THIS SUBROUTINE COMPRESSES THE STATEMENT STRING,
624  * THE INPUT CONSISTS OF A FROM POINTER AND A TO
625  * POINTER, IN START AND ENDD RESPECTIVELY
626  * ON OUTPUT, ENDD IS UPDATED TO POINT ONE BEYOND
627  * THE NEW END, AND INDEX 2 CONTAINS THE INPUT STAR
628  * VALUE.
629  *
630  MOVE  DC       *-*        RETURN ADDRESS
631  SAVE  LDX  L3  *-*        STRING TO MOVE, MODIFIABLE
632  ENDD  EQU      * 1        LABEL NEXT INSTRUCTION ADD
633        LDX  L2  *-*        MOVE AREA START, MODIFIABL
634        LD       ONE
635        S    L   START      1-START
636        STO      TEST 1     SAVE FOR END TEST
637  MOVE1 LD     3 0          NEXT WORD TO MOVE
638        STO    2 0          COMPRESS STRING
639  TEST  MDX  L3  *-*        END MOVE TEST, MODIFIABLE
640        MDX      DONE       DONE IF NOT ZERO OR NEG
641        MDX  I3  START      EFFECTIVE INCR MOVE ADDR
642        MDX      *          NO-OP FOR POSSIBLE SKIP
643        MDX    2 1          INCR MOVE TO ADDRESS
644        MDX      MOVE1      NEXT WORD
645  DONE  STX    2 ENDD       NEW END OF STRING
646        LDX  I2  START      RESET INDEX REG 2
647        BSC  I   MOVE       EXIT
648  *
649  * THIS ROUTINE COMPLETES STRING COMPRESSION AND
650  * EXITS TO THE NEXT PHASE
651  *
652  EXIT  LD       ENDD       END OF STRING
653        BSC  L   EXIT1, -   IF ENDD IS ZERO, NO MOVE
654        BSI      MOVE       MOVE THE REST OF THE STRIN
655        LD       ENDD
656        STO  L   EOFS       NEW END OF STRING ADDR
657  EXIT1 BSI  L   ROLRX      CALL DOWN PHASE 10
658        DC       10               NEXT PHASE NUMBER
659  *
```

```
660   * THIS SUBROUTINE CALLS THE GET CONSTANT SUBROUTIN
661   * TO FIND A SUBSCRIPT INTEGER, IT TESTS THE GCON
662   * OUTPUT TO DETERMINE VALIDITY OF THE SUBSCRIPT,
663   *
664   GCONS DC       *-*          RETURN ADDRESS
665         MDX  L   K,1          INCR CONSTANT INDEX
666         BSI      GCON         GET CONSTANT
667         LD   L   CONL         CONSTANT LENGTH
668         S        ONE          ONE
669         BSC  L   ERR6,Z       ERROR IF NOT AN INTEGER
670         LD    1  -1           CONSTANT
671         S        ONE          CONSTANT = 1
672         LDX  I2  K            J K
673         BSC  I   GCONS,-      EXIT IF POSITIVE INTEGER
674   ERR6  LDX  X1  D6           ILLEGAL SUBSCRIPT
675         MDX      ERR          SET UP ERROR 24
676   *
677   PLUS  DC       /0E            SIGN
678   *
679   * THIS SUBROUTINE FINDS THE NEXT CONSTANT IN THE
680   * STRING, IF ANY, ON INPUT, K POINTS TO THE
681   * EXPECTED FIRST WORD OF THE CONSTANT, AND INDEX 1
682   * TO THE NEXT AVAILABLE WORD IN THE WORK AREA,
683   * ON EXIT, THE CONSTANT IS IN THE WORK AREA, INDEX
684   * 1 UPDATED, AND THE FOLLOWING SWITCHES SET,
685   *                CONL 0          NO CONSTANT FOUND
686   *                     OTHER      LENGTH OF CONSTANT
687   *                CONTP 0         REAL CONSTANT
688   *                      1         INTEGER CONSTANT
689   *                      2         HOLLERITH CONSTANT
690   *
691   * IN ADDITION, K IS UPDATED TO POINT TO THE NEXT
692   * WORD PAST THE CONSTANT
693   *
694   GCON  DC       *-*          RETURN ADDRESS
695         SRA      16           CLEAR ACCUMULATOR
696         STO  L   CONL         ZERO CONSTANT LENGTH
697         STO  L   CONTP        CONSTANT TYPE
698         STO      NEG          NEGATIVE SWITCH
699         STX  L3  SAV3 1       SAVE INDEX REGISTER 0
700         LDX  I3  K            SET INDEX REG 3   K
701         LD    3  0            NEXT STMNT WORD
702         S        PLUS         PLUS SIGN
703         BSC  L   * 2,Z        BR IF NOT PLUS SIGN
704         MDX   3  1            DISCARD PLUS SIGN
705         MDX      GCONA        BR TO LOAD NEXT STMNT WORD
706         LD    3  0            NEXT STMNT WORD
707         S        MINUS        MINUS SIGN
708         BSC  L   * 2,Z        BR NOT MINUS SIGN CHAR
709         STX      NEG          MINUS SIGN, SET NEG NONZER
710         MDX   3  1            INCR STMNT POINTER
711   GCONA LD    3  0            NEXT STMNT WORD
712         S        BCD          QUOTE MARK
713         BSC  L   GCON1,Z      BR IF NOT HOLLERITH
714         LD       NEG          NEGATIVE INDICATOR
715         BSC  L   * 2, -       BR IF POSITIVE
716   ERR10 EQU      *            LABEL FOR NEXT INSTRUCTION
717         LDX  X1  D10          NEGATIVE HOLLERITH CONSTAN
718         MDX      ERR          BR TO SET UP ERROR 75
719         MDX  L   CONTP,2      SET DATA TYPE  2
```

```
720           MDX    3 1            INCR STMNT POINTER
721   *
722   *                 PROCESS EBCDIC CONSTANTS
723   *
724   GCON2   SLT       16           CLEAR EXTENSION
725           LD     3 0             NEXT STMNT WORD
726           EOR       BCD          QUOTE MARK CONSTANT
727           BSC  L    GCON4, -     BR IF QUOTE MARK
728           SRA       8
729           BSC  L    ERR11, -     BR IF ILLEGAL HOLLERITH CO
730           LD     3 0             NEXT STMNT WORD
731   B       MDX    3 1             INCR STMNT POINTER
732           RTE       16           BRING IN PREVIOUS WORD
733           BSC  L    GCON2 1, -   DISCARD IF EMPTY
734           STO    1 0             WORK AREA
735           BSI  L    INCRP        INCR WORK AREA POINTER
736           LD     1 -1            LOAD LAST O/P WORD
737           AND       LFF          MASK OUT LEFT CHARACTER
738           MDX  L    CONL,1       INCR CONSTANT LENGTH IF
739           BSC  L    GCON2 1,Z    WORD FULL,
740           LD     1 -1            LAST WD FROM WORK AREA
741           SRA       8            RIGHT JUSTIFY LEFT CHAR
742           SLT       8            PACK IN ONE MORE CHARACTER
743           STO    1 -1            RESTORE TO WORK AREA
744           MDX       GCON2 1      GET NEXT INPUT
745   *
746   BCD1    DC        /7D00        LEFT ADJUSTED QOUTE MARK
747   LFF     DC        /FF          LOW 8 BIT MASK
748   *
749   GCON4   MDX    3 1             INCR STMNT POINTER
750           LD     3 0             NEXT STMNT WORD
751           S         BCD          QUOTE MARK
752           BSC  L    QT1,Z        BR IF NOT ANOTHER QUOTE
753           LD        BCD1         COMPRESS TO ONE QUOTE
754           MDX       B            CONTINUE PACKING
755   QT1     SLT       16           SHIFT IN PREVIOUS WORD
756           BSC  L    QT2, -       BR IF ZERO
757           OR   L    BLANK        BLANK LOW CHAR IF ZERO
758           STO    1 0             WORK AREA
759           BSI  L    INCRP        INCR WORK AREA POINTER
760           MDX  L    CONL,1       UPDATE CONSTANT LENGTH
761   QT2     LD   L    CONL         CONSTANT LENGTH
762           BSC  L    GCONX,Z      BR TO EXIT GCON ROUTINE
763   ERR11   LDX  X1   D11          ILLEGAL CONSTANT LENGTH
764           BSC  L    ERR          SET UP ERROR 77
765   *
766   REAL    DC        /5E          REAL CONSTANT INDR
767   LNGTH   DC        0            LENGTH OF REAL INDR
768   NEG     DC        *-*          NEGATIVE CONSTANT HDR
769   *
770   *                 PROCESS REAL CONSTANTS
771   *
772   GCON1   LD     3 0             NEXT STMNT WORD
773           S         REAL         REAL CONSTANT INDR
774           BSC  L    GCON3,Z      BR IF NOT A REAL CONSTANT
775           LD     3 1             FIRST WORD OF REAL CONSTAN
776           STO    1 0             WORK AREA
777           STO       WD           SAVE IN CASE NEGATIVE
778           LD     3 2             SECOND WORD OF REAL CON
779           STO    1 1             WORK AREA
```

```
780           LD      LNGTH     LENGTH OF REAL INDR
781           STO  L  CONL      SET CONSTANT LENGTH
782           S    L  THREE     THREE
783           STO     EXTND     NORMAL PRECISION  NON-ZERO
784           BSC  L  INC3,Z    BR IF NORMAL PRECISION
785           LD    3 2         2ND WORD OF EXTENDED REAL
786           STO     WD        SAVE IN CASE NEGATIVE
787           LD    3 3         3RD WORD OF EXTENDED REAL
788           STO   1 2         WORK AREA
789           MDX   3 1         SPECIAL INCR FOR 3RD WORD
790           BSI  L  INCRP     INCR WORK AREA POINTER
791   INC3    MDX   3 3         UPDATE STMNT POINTER
792           BSI  L  INCRP     INCREMENT WORK AREA POINTE
793           BSI  L  INCRP     TWICE,
794           LD      NEG       NEGATIVE CONSTANT INDR
795           BSC  L  GCONX, -  EXIT IF POSITIVE
796           LD    3 -1        LAST WORD  THIRD OR SECOND
797           MDX  L  EXTND,0   SKIP NEXT IF EXTENDED PREC
798           AND     HFF00     MASK OUT EXPONENT
799           STO     WD 1      SAVE FOR COMPLEMENT
800           EOR   3 -1        IF NORMAL, SET EXPONENT
801           STO     EXP       IF EXTENDED, SET ZERO
802           SRT     24        CLEAR ACC AND EXT
803           SD      WD        COMPLEMENT MANTISSA
804           STO   1 -2        LAST-1  WORD TO WORK
805           RTE     16        LAST WORD
806           OR      EXP       SET EXPONENT IF NORMAL PRE
807           STO   1 -1        LAST WORD TO WORK
808           MDX     GCONX     EXIT
809   *
810   WD      BSS  E  2         TEMP FOR COMPLEMENTING
811   EXTND   DC      0         NORMAL PRECISION NONZERO
812   HFF00   DC      /FF00     HIGH ORDER 8-BIT MASK
813   EXP     DC      0         EXPONENT FOR NORMAL PREC
814   *
815   *               PROCESS INTEGER CONSTANTS
816   *
817   GCON3   LD    3 0         LOOK FOR PARTIAL WORD SPEC
818           AND     H7E00     EXTRACT 1ST CHAR OF SYMBOL
819           EOR     SYMZ      CHECK FOR A Z
820           BSC  L  BPWS, -   BRANCH IF SO,
821           MDX     * 5       BR TO CHECK FOR OPERATOR
822           LD    3 0         LOAD CHARACTER
823           EOR  L  LPAR      CHK FOR LEFT PARENTHESIS
824           BSC  L  FPWS, -   BR IF LEFT PARENTHESIS
825           LD    3 0         CHK FOR OPERATOR
826           BSC  L  ERR8,-    BRANCH IF SO  ERROR ,
827           BSI  L  GCONI     GET INTEGER
828           BSI  L  INCRP     INCR WORK AREA POINTER
829           MDX     GCONX     EXIT
830   FPWS    LD      NEG       NEGATIVE CONSTANT INDICATO
831           BSC  L  ERR10,Z   BR IF NEG PARTIAL WORD
832           SRA     16        CLEAR ACCUMULATOR
833           STO     PW        CLEAR PARTIAL WD ACC
834           STO     PWTOT     CLEAR PARTIAL WORD COUNT
835   *
836   *               PROCESS PARTIAL WORD CONSTANTS
837   *
838   GCON9   MDX   3 1         DISCARD DELIMITER
839           BSI  L  GCONI     GET INTEGER
```

```
840           LD    1 0          COUNT FROM WORK AREA
841           STO     PWCT       SAVE COUNT FOR THIS FIELD
842           A       PWTOT      UPDATE TOTAL COUNT
843           STO     PWTOT      RESTORE
844           S       L16        DECIMAL 16
845           BSC   L * 3,       BR PRATIAL WORD COUNT O<
846   ERR12   LDX   X1 D12       PARTIAL WD CNT GT 16
847           BSC   L ERR        BR TO SET UP ERROR 79
848           LD    L CONL       CONSTANT LENGTH
849           BSC   L ERR12, -   BR CONSTANT NOT FOUND
850           LD     3 0         NEXT STMNT WORD
851           S       EQSGN      EQUAL SIGN
852           BSC   L ERR12,Z    BR NO EQUAL SIGN
853           MDX    3 1         INCR STMNT POINTER
854           BSI   L GCONI      GET INTEGER
855           STX    2 SAV2 1    SAVE INDEX REG 2
856   PWCT    EQU     * 1        PARTIAL WD ELEMENT LENGTH
857           LDX   L2 *-*       GET SHIFT COUNT, MODIFIABL
858           LD     1 0         HEADER WD FROM WORK AREA
859           SRT    2 0
860           BSC   L ERR12,Z    BR IF CONSTANT TOO LARGE
861           LD      PW         PARTIAL WORD TO ACCUMULATO
862           SLT    2 0         PUT IN NEW FIELD
863   SAV2    LDX   L2 *-*       SET XR2, MODIFIABLE
864           STO     PW         STORE UPDATED PARTIAL WORD
865           LD     3 0         NEXT STMNT WORD
866           S     L COMMA      COMMA CONSTANT
867           BSC   L GCON9, -   BR TO GET NEXT FIELD IF AN
868           LD     3 0         NEXT STMNT WORD
869           S     L RPAR       RIGHT PARENTHESIS CONSTANT
870           BSC   L ERR12,Z    ERROR IF NOT RT PARENTHESI
871           LD      PW         PARTIAL WORD
872           STO    1 0         TO WORK AREA
873           BSI   L INCRP      INCR WORK AREA POINTER
874           MDX    3 1         INCR STMNT POINTER
875           STX   L3 K         SAVE AS K
876           LDX    3 1         SET INTEGER SWITCH
877           STX   L3 CONTP     CONSTANT TYPE
878           STX   L3 CONL      CONSTANT LENGTH
879           MDX      SAV3      BR TO RESTORE XR3 AND EXIT
880   GCONX   STX   L3 K         SAVE K
881   SAV3    LDX   L3 *-*       RESTORE XR3, MODIFIABLE
882           BSC   I GCON       EXIT
883   *
884   *                CONSTANTS AND WORK AREA
885   *
886   PW      DC       *-*       PARTIAL WORD TEMPORARY
887   PWTOT   DC       *-*       PARTIAL WORD TOTAL LENGTH
888   L16     DC       16        PARTIAL WD COUNT LIMIT
889   H7E00   DC       /7E00     MASK TO GET 1ST CHAR
890   SYMZ    DC       /5200     Z DENOTES HEX DIGITS FOLLO
891   PW1     BSS   E  1         HOLDS 1-4 CHARACTERS  HEX
892   PW2     DC       *-*       DIGITS W/ Z PRECEDING ,
893   HC000   DC       /C000     MASK TO GET ZONE BITS
894   HF000   DC       /F000     MASK TO GET NUMERIC PART
895   H9000   DC       /9000     GT THAN WHICH ARE CHARS
896   H1800   DC       /1800     ADD TO NO, PART TO GET BIN
897   HFC00   DC       /FC00     MASK TO GET 1ST CHAR
898   EQSGN   DC       /3E
899   *
```

```
900   *                CONTINUE PROCESSING PARTIAL WD CONS
901   *
902   BPWS  LD    L  NEG          CHK FOR NEGATIVE
903         BSC   L  ERR10,Z      BRANCH IF SO  ERROR ,
904         STO      PW           CLEAR PARTIAL WORD
905         STO      PW2          ZERO 2ND HALF OF SYMBOL
906         LD    3  0            CHECK FIRST CHAR AFTER
907         SLA      7            THE Z, MUST BE NON-BLANK,
908         AND      HFC00        EXTRACT THE CHAR
909         BSC   L  ER14, -      BR TO SET UP ERROR IF BLAN
910         SLT      32           CLEAR EXTENSION
911         LD    3  0            GET 1ST HALF
912         SLA      1            SHIFT OUT INDICATOR BIT
913         STO      PW1          SAVE IN PW1 TEMPORARY
914         LD    3  1            CHECK FOR 3-5 CHARS
915         BSC   L  SYMGT,-      BRANCH IF NOT
916         SLA      1            SHIFT OUT INDICATOR BIT
917         SRA      1            REPLACE IT W/ ZERO, MOVE
918         SRT      14           LAST 15 BITS TO 0, TO LEAV
919         OR       PW1          THE CHARACTERS CONTIGUOJS
920         STO      PW1          AND LEFT JUSTIFIED IN
921         RTE      16           PW1 AND PW2,
922         STO      PW2
923         MDX   3  1            INCREMENT STRING POINTER
924         LD    3  1            CHECK TO BE SURE THE NEXT
925         BSC   L  ERR12, Z     CHAR IS AN OPERATOR, BR NO
926   SYMGT STX   2  X2S 1        SAVE X2
927         LDX   2  6            SET SHIFT COUNTER FOR 1ST
928   GTHD  LDD      PW1          LOAD THE PW SPEC ZHHHH
929         SLT   2  0            GET THE NTH CHARACTER
930         AND      HC000        CHK ZONE BITS
931         BSC   L  CKAD, -      BR IF A TO F,
932         S        HC000        ELSE IT MUST BE NUMERIC,
933         BSC   L  ER14,Z       ERROR IF NOT,
934         LDD      PW1          GET THE DIGIT, IN 6-BIT
935         SLT   2  0            CODE, LEFT JUSTIFY IN ACC,
936         SLA      2            STRIP ZONE BITS
937         AND      HF000        DROP OTHER DIGITS
938         BSC   L  PTHD, -      BRANCH ON DIGIT ZERO
939         S        H9000        CHK WITHIN RANGE 0 TO 9
940         BSC   L  ER14,-Z      BR NOT 0 TO 9
941         A        H9000         RESTORE ACTUAL VALUE
942   PTHD  SRT      16           LEFT JUSTIFY DIGIT IN EX-
943         LD       PW           TENSION, THEN APPEND IT TO
944         SLT      4            THE PARTIAL WD BEING BUILT
945         STO      PW
946         MDX   2  6            INCR XR2 TO GET NEXT CHAR
947         MDX      GTHD         BR TO GET CHARACTER
948   CKAD  LDD      PW1          CHECK DIGITS A-F,
949         SLT   2  0            GET THE CHARACTER
950         AND      HFC00        DROP OTHER CHARACTERS
951         BSC   L  ENDHD,       EXIT IF 000000
952         S        H1800        CHECK TO SEE IF IN RANGE
953         BSC   L  ER14,-Z      BRANCH IF NOT A-F,
954         A        HFC00        ADD BACK 1800 PLUS 2400 TO
955         SLA      2            GET BINARY EQUIVALENT,
956         MDX      PTHD         INSTALL IN PARTIAL WORD
957   ENDHD LD       PW           PARTIAL WORD TEMPORARY
958         STO   1  0            STORE PARTIAL WD
959         BSI   L  INCRP        INCR WORK AREA POINTER
```

```
960          MDX    3 1           K K 1
961          STX   L3 K           SAVE K
962          LDX    3 1
963          STX   L3 CONTP       CONSTANT TYPE
964          STX   L3 CONL        CONSTANT LENGTH
965   X2S    LDX   L2 *-*         RESTORE XR2, MODIFIABLE
966          MDX      SAV3        BR TO RESTORE XR3 AND EXIT
967   ER14   LDX    1 D14         ERROR - CHAR NOT A-F IN
968          BSC   L  ERR         PARTIAL WORD SPEC,
969   *
970   * THIS SUBROUTINE IS USED BY GCON TO GET AN INTEGE
971   * CONST WHICH IS PUT IN NEXT AVAIL WORK AREA WORD,
972   * WK AREA POINTER IS UPDATED,
973   *
974   GCONI  DC        *-*         RETURN ADDRESS
975          SLT       32         CLEAR ACC AND EXTENSION
976          STO    1  0          CLEAR NEXT WORK AREA WORD
977          LD     3  0          NEXT STMNT WORD
978          STD       NOS        SAVE INTEGER WORD 1
979          MDX    3  1          INCR STMNT POINTER
980          LD     3  0          NEXT STMNT WORD
981          BSC   L  GCON6,-     BR IF NO WORD 2
982          SLA       1          KNOCK OUT HIGH BIT
983          STO       NOS 1      SAVE INTEGER WORD 2
984          MDX    3  1          INCR STMNT POINTER
985   GCON6  LDS       0          CLEAR CARRY BIT
986          LD        NOS        INTEGER TEMPORARY
987   GCON7  SLA       2
988          BSC   L  * 3,C       BR IF NOT FINISHED
989          BSC   L  ERR6,Z      BR TO ERROR IF NOT A DIGIT
990          MDX      GCON8       BR FINISHED
991          BSC   L  ERR6,-      BR TO ERROR IF NOT A DIGIT
992          SLA       1          STRIP OFF ZONE BITS
993          SRA       12         ISOLATE DIGIT
994          STO   L  NOST        SAVE DIGIT
995          S         NINE       DIGIT LIMIT VALUE
996          BSC   L  ERR6,-Z     BR IF NOT A DECIMAL DIGIT
997          LD     1  0          ACCUMULATED TOTAL
998          M         TEN        MPY BY 10
999          SLT       1
1000         BSC   L  ERR6,Z      BR IF VALUE TOO LARGE
1001         SLT       15         SHIFT INTO ACCUMULATOR
1002         A     L   NOST       ADD LATEST DIGIT
1003         BSC   L  ERR6,O      BR IF VALUE TOO LARGE
1004         STO    1  0          UPDATE ACCUMULATED TOTAL
1005         LDD       NOS        INTEGER TEMPORARY
1006         SLT       6          SHIFT OFF PROCESSED DIGIT
1007         STD       NOS        RESTORE
1008         MDX       GCON7      BR TO FIND NEXT DIGIT
1009   GCON8 S        1  0          NEGATE INTEGER
1010         MDX   L  NEG,0       SKIP NEXT IF NEG INDR OFF
1011         STO    1  0          SET INTEGER NEGATIVE
1012         MDX   L  CONL,1      INCR CONSTANT LENGTH
1013         MDX   L  CONTP,1     INCR CONSTANT TYPE
1014         BSC   I  GCONI       EXIT
1015   *
1016   *              CONSTANTS, WORK, AND PATCH AREA
1017   *
1018   TEN    DC        10         DECIMAL 10 MPY CONSTANT
1019   NOS    BSS   E   2          INTEGER TEMPORARY STORE
```

```
1020  NINE  DC      9           DECIMAL 9 DIGIT LIMIT CON
1021  PATCH BSS     50          PATCH AREA
1022  WORK  BSS     OVERL-*+320*4   WORK AREA FOR DATA
1023  WKEND EQU     *           END OF WK AREA   1
1024  *
1025        END     BEGIN
```