```
1   *    GENERAL AUTOMATION, INC.  ALL RIGHTS RESERVED
2   ***********************************************
3   *
4   *    PROGRAM NAME  FORTRAN PH-02
5   *
6   *    MODEL NUMBER  8F002
7   *
8   *    PURPOSE       FORTRAN PHASE-02
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        RPH INITIAL RELEASE
18  *
19  ***********************************************
20  ***********************************************
21  * GA 18/30  FORTRAN COMPILER   05/01/70
22  *STATUS - VERSION 1, MODIFICATION 0
23  *FUNCTION/OPERATION-
24  *    * DETERMINES THE STMNT TYPE FOR EACH STMNT,
25  *      INSERTS THE TYPE CODE INTO THE STMNT ID WORD
26  *    * PLACES THE TERMINAL CHARACTER AT THE END OF
27  *      EACH STATEMENT
28  *    * CONVERTS SUBPROGRAM NAMES LONGER THAN FIVE
29  *      CHARACTERS TO FIVE-CHARACTER NAMES
30  *    * CONVERTS FORTRAN-SUPPLIED SUBPROGRAM NAMES
31  *      ACCORDING TO THE SPECIFIED PRECISION
32  *ENTRY POINTS-
33  *    * START - PHASE 2 IS LOADED TO CORE BY *ROLR
34  *              VIA CALL TO ROLRX FROM PHASE 1,
35  *              START IS THE LABEL ON THE XEQ ADDR.
36  *INPUT-
37  *    NONE IN THE USUAL SENSE, HOWEVER, THE STMNT
38  *    STRING SCANNED BY THE COMPILER MAY BE
39  *    CONSIDERED INPUT
40  *OUTPUT-
41  *    NONE IN THE USUAL SENSE, HOWEVER, THE STMNT
42  *    STRING ALTERED BY THE COMPILER MAY BE
43  *    CONSIDERED OUTPUT
44  *EXTERNAL ROUTINES-N/A
45  *EXITS-
46  *    NORMAL-
47  *       THE PHASE EXITS TO ROLRX, THE INTERPHASE
48  *       ROLLER, FOR BOTH NORMAL AND ERROR EXITS.
49  *    ERROR-
50  *       THE ONLY ERROR WHICH INTERRUPTS THE NORMAL
51  *       FLOW OF COMPILATION IS THE OVERLAPPING OF
52  *       THE STRING AND SYMBOL TABLE.  WHEN THE
53  *       OVERLAP CONDITION IS DETECTED, THE PHASE
54  *       IS EXITED BY FIRST SETTING AN INDICATOR
55  *       WORD,ERROR TO AN EVEN VALUE AND SECOND,BY
56  *       CALLING PHASE 3 VIA ROLRX.  EACH SUCCEED-
57  *       ING PHASE DETECTS THIS ERROR AND IMMEDIATE-
58  *       LY EXITS THE PHASE THUS PASSING CONTPOL TO
59  *       SUCCEEDING PHASES UNTIL PHASE 21,  AT THAT
```

```
60  *       POINT A MESSAGE IS PRINTED,
61  *TABLES/WORK AREAS-
62  *    NONE EXCEPT STATEMENT STRING AREA, SYMBOL
63  *    TABLE, AND FORTRAN COMMUNICATION AREA
64  *ATTRIBUTES-N/A
65  *NOTES-
66  *       THE ERROR DETECTED BY THIS PHASE IS NUM 4
67  *    THE SWITCHES USED IN PHASE 2 FOLLOW, IF NON-
68  *    ZERO, THE SWITCH IS TRANSFER=T, IF ZERO, THE
69  *    SWITCH IS NORMAL=N,
70  *       * SW1-REAL OR INTEGER INDICATOR
71  *            T=STATEMENT IS REAL OR INTEGER
72  *       * SW2-DO INDICATOR
73  *            T=STATEMENT IS DO
74  *       * SW3-WHICH HALF OF WORD IS CHARACTER
75  *            T=RIGHT HALF
76  *       * SW4-CONTENTS OF *NAMS*
77  *            T=CHARACTER IN NAMS
78  *       * SW5-TYPE OF ARGUMENT TO MAKE CHARACTER
79  *            T=OPERATOR
80  *       * SW6-POSITION IN STATEMENT
81  *            T=END OF STATEMENT
82  *       * SW7-DATA INDICATOR
83  *            T=STATEMENT IS DATA
84  *       * SW8-QUOTE ENCOUNTERED
85  *            N=NOT ENCOUNTERED
86  *       * SW9-OUTPUT WHICH HALF OF WORD
87  *            N=LEFT HALF
88  *            T=RIGHT HALF
89        ABS
90  *     SYSTEM AND FORTRAN EQUATES
91  MEMRY EQU    /7FFF      MAXIMUM CORE SIZE
92  PHSIZ EQU    4*320      MAXIMUM PHASE SIZE
93  OVERL EQU    MEMRY-PHSIZ    PHASES 2-29 START
94  FCOM  EQU    OVERL-22   FORTRAN COMM, TABLE
95  PHNTB EQU    FCOM-56    PHASE TABLE
96  ROLRX EQU    PHNTB-50   INTERPHASE CALL
97  *     FORTRAN COMMUNICATION AREA
98        ORG    FCOM
99  SOFS  BSS    1          START OF STRING
100 EOFS  BSS    1          END OF STRING
101 SOFST BSS    1          START OF SYMBOL TABLE
102 SOFNS BSS    1          START OF NON-STMNT NUMBERS
103 SOFXT BSS    1          START OF SUBSCRIPT TEMPS
104 SOFGT BSS    1          START OF GENERATED TEMPS
105 EOFST BSS    1          END OF SYMBOL TABLE
106 COMON BSS    1          NEXT AVAILABLE COMMON
107 CSIZE BSS    1          SIZE OF COMMON
108 ERROR BSS    1          OVERLAP ERROR
109 FNAME BSS    1          PROGRAM NAME
110       BSS    1          2ND WORD OF NAME
111 SORF  BSS    1          SUBR (*) OF FUNC (*)
112 CCWD  BSS    1          CONTROL CARD WORD
113 *                  BIT 15 TRANSFER TRACE
114 *                  BIT 14 ARITHMETIC TRACE
115 *                  BIT 13 EXTENDED PRECISION
116 *                  BIT 12 LIST SYMBOL TABLE
117 *                  BIT 11 LIST SUBPROGRAM NAMES
118 *                  BIT 10 LIST SOURCE PROGRAM
119 *                  BIT  9 ONE WORD INTEGERS
```

```
120  IOCS  BSS     1          IOCS CONTROL CARD WORD
121  *                   BIT 15 CARD
122  *                   BIT 14 PAPER TAPE
123  *                   BIT 13 TYPEWRITER
124  *                   BIT 12 1443 PRINTER
125  *                   BIT 11 MAGNETIC TAPE
126  *                   BIT 10 KEYBOARD
127  *                   BIT  9 1442 PUNCH
128  *                   BIT  8 DISK
129  *                   BIT  7 NOT USED
130  *                   BIT  6 NOT USED
131  *                   BIT  5 NOT USED
132  *                   BIT  4 NOT USED
133  *                   BIT  3 PLOTTER
134  *                   BIT  2 NOT USED
135  *                   BIT  1 UNFORMATTED DISK
136  *                   BIT  0 UNFORMATTED TAPE
137  DFCNT BSS     1          DEFINE FILE COUNT
138  LCOMN BSS     2          INSKEL COMMON SIZE
139  ICCER BSS     2          IOCS CONTROL CARD ERROR
140        BSS     2          SYSTEM LOADER USE
141  *                  END OF FORTRAN COMMUNICATION
142  *                  AREA
143  *                  CHECKS THE FORTRAN COMMUNICATIO
144  *                  AREA ERROR WORD FOR AN OVERLAP
145        ORG     OVERL
146  START LD      ERROR      CHECK FOR OVERLAP ERROR
147        BSC  L  BACK,+-    BRANCH IF NOT
148  *               TRANSFERS TO ROLRX ROUTINE
149  WAIT  BSI  L  ROLRX      GO GET NEXT
150        DC      03         NEXT PHASE NUMBER
151  *                  MOVES THE ENTIRE STATEMENT
152  *                  STRING NEXT TO THE SYM TBL
153  BACK  LD      EOFS       LOAD END OF STRING ADDRES
154        S       SOFS       SUBTRACT START OF STRING
155        STO     *+1        TO GET SIZE OF STRING - 1
156        LDX  L3 *-*        LOAD SIZE IN XR3
157        MDX   3 1          GET SIZE OF STRING
158        LDX  I2 EOFS       INITIALIZE POINTERS FOR
159        LDX  I1 EOFST      MOVING STRING UP TO SYMBO
160  BACK1 LD    2 0          TABLE
161        STO   1 +2         MOVE WORD FROM STRING TO
162        MDX   1 -1         SYMBOL TABLE - MOVE
163        MDX   2 -1         POINTERS - DECREMENT
164        MDX   3 -1         COUNTER OF RANGE OF LOOP
165        MDX     BACK1      CONTINUE LOOP
166        MDX   1 3          RESET INPUT POINTER
167        LDX  I2 SOFS       RESET OUTPUT POINTER
168        BSC  L  CQCT       CHECK FOR NONPROCESS,PUNC
169  *                  MOVES STMNTS FROM THE INPUT
170  *                  STRING TO THE OUTPUT STRING
171  MOVE  LD    1 0          LOAD STATEMENT ID WORD
172        AND     S07FC      ZERO ALL BUT NORM
173        SRA     2          MOVE NORM TO RIGHT
174        STO     XTE1       SET NORM COUNTER
175  MXS   LD    1 0          LOAD ID WORD
176        BSI  L  PUT        PUT ON OUTPUT STRING
177        MDX   1 1          MOVE INPUT POINTER
178        MDX  L  XTE1,-1    DECREMENT NORM COUNTER
179        MDX     MXS        CONTINUE LOOP
```

```
180   *                          MOVE TO NEXT STATEMENT
181   MOVE1 LDX   L1 *-*         MOVE INPUT POINTER TO
182   IDSV  EQU      *-1         NEXT STATEMENT
183   *                          INITIALIZES FOR A SCAN OF
184   *                          THE STMNT STRING
185   ZA0   LD     1 0           LOAD STATEMENT ID WORD
186         SLA      5           SHIFT TO GET NORM
187         SRA      7
188         STO      *+1
189   ZA0A  LDX   L3 *-*         SAVE NORM IN XR3
190         STX    1 IDSV        SAVE INPUT POINTER
191         LD       ZA0A+1      ADD NORM TO POINTER TO GE
192         A        IDSV        ADDRESS OF NEXT STATEMENT
193         STO      IDSV
194         LD     1 0           LOAD ID WORD
195         BSC      E           DOES STMNT HAVE STMNT NO,
196         MDX    3 -2          YES, MODIFY NORM CTR
197         MDX    3 -1          MODIFY CTR SO HAVE SIZE O
198         NOP                  STMNT WITHOUT ID WORD AND
199         STX   L3 NCNT        STATEMENT NUMBER
200         STX   L3 NCNTS       SAVE MODIFIED NORM
201   *                          CHECK FOR SPECIAL END INDICATOR
202   ZA1   LD     1 0           LOAD STATEMENT ID WORD
203         SRA      11          GET ID TYPE CODE
204         S        ENDC        CHECK FOR END TYPE CODE
205         BSC   L  ZA1B,Z      BRANCH IF NOT END
206         LD    L  ERID        LOAD ERROR STMNT ID WORD
207         BSI   L  PUT         PUT WORD ON OUTPUT STRING
208         LD       ERNO2       LOAD ERROR NUMBER
209         BSI   L  PUT         PUT NO, ON OUTPUT STRING
210   *                          PLACES THE END STMNT ID WORD
211   *                          ON THE STRING
212   ZA1A  STX   L2 EOFS        SET END OF STRING ADDRESS
213         LD       S1000       LOAD END STMNT ID WORD
214         BSI   L  PUT         PUT WORD ON OUTPUT STRING
215         BSC   L  CALLC       GO TO PHASE EXIT
216   *                          CHECKS FOR ARITHMETIC STMNTS
217   ZA1B  S        ATHC        HAS STMNT BEEN CLASSIFIED
218         BSC   L  MOVE,Z      YES, GO CHECK STMNT
219   *                          INITIALIZE GET SUBROUTINE
220         STX    1 IDSV1       SAVE INPUT POINTER
221         LD     1 0           LOAD STMNT ID WORD
222         BSC      E           HAS STMNT NO,
223         MDX    1 2           YES, MOVE POINTER OVER NO
224         MDX    1 1           MOVE POINTER TO FIRST WOR
225         STX    1 IDSA1       AFTER ID = SAVE ADDRESS
226         SLA      16
227         STO   L  SW3         CLEAR SWITCH 3
228         STO   L  SW6         CLEAR SWITCH 6
229         STO      SW7         CLEAR SWITCH 7
230         STO      SW8         CLEAR SWITCH 8
231         STO      SW9         CLEAR SWITCH 9
232   *                          CHECKS THE ARITHMETIC STMNT
233   *                          FOR THE VARIOUS TYPES OF OPTRS
234   ZA2   BSI   L  GET1        GET CHAR FROM INPUT STRIN
235         STO      X           SAVE CHARACTER
236         S        SLHHS       IS CHARACTER SLASH
237         BSC   L  ZA5,+-      BRANCH IF YES
238         LD       X
239         S        EQUC        IS CHARACTER EQUAL SIGN
```

```
240          BSC  L  ZA3,*-      BRANCH IF YES
241          LD      X           IS CHARACTER ZERO
242          BSC  L  ZA2,Z       BRANCH IF NOT
243          MDX     ZA5         BRANCH = END OF STMNT
244  ZA3     BSI  L  GET1        GET CHAR FROM INPUT STRIN
245          STO     X           SAVE CHARACTER
246          S       CMAC        IS CHARACTER COMMA
247          BSC  L  ZA5,+-      BRANCH IF YES
248          S       LPC         IS CHAR LEFT PARENTHESIS
249          BSC  L  ZA4,+-      BRANCH IF YES
250          LD      X
251          BSC  L  ZA3,Z       BRANCH IF NOT END OF STMN
252  *                       INITIALIZE  GET SUBROUTINE
253  ZA4     LDX  L1 *-*         LOAD STATEMENT ID WORD
254  IDSA1   EQU     *-1
255          STO  L  SW3         CLEAR SWITCH 3
256          STO  L  SW4         CLEAR SWITCH 4
257          STO  L  SW6         CLEAR SWITCH 6
258          LD   L  NCNTS       RESET NORM TO
259          STO  L  NCNT        ORIGINAL VALUE
260  *                       PUT ARITHMETIC ID WORD IN IDWD
261          SLA     16          GET ARITHMETIC ID WORD
262          STO     IDWD        STORE IN ID WORD
263          MDX     ZA6         GO PUT ON OUTPUT STRING
264  *                       CONSTANTS
265  ERNO2   DC      3           ERROR NO, 3
266  S07FC   DC      /07FC       NORM MASK
267  XTE1    DC      0           NORM COUNTER
268  ENDC    DC      /02         END STMNT ID TYPE
269  S1000   DC      /1004       END STMNT ID WORD
270  ATHC    DC      /00-/02     ARITHMETIC STMNT ID TYPE
271  IDSV1   DC      0           INPUT POINTER
272  EQUC    DC      /7E         EQUAL SIGN
273  CMAC    DC      /6B         COMMA
274  LPC     DC      /4D-/6B     LEFT PARENTHESIS
275  IDWD    DC      0           ID WORD
276  REALC   DC      /4800       REAL STMNT ID TYPE
277  INTC    DC      /5000-/4800   INTEGER STMNT ID TYPE
278  DOC     DC      /5800-/5000   DO STMNT ID TYPE
279  EIGHT   DC      8           EIGHT
280  SW7     DC      *-*         SWITCH 7
281  SW8     DC      *-*         SWITCH 8
282  SW9     DC      *-*         SWITCH 9
283  SW1     DC      0           SWITCH 1
284  SW2     DC      0           SWITCH 2
285  X       DC      0           TEMPORARY STORAGE
286  SLHHS   DC      /61         SLASH
287  ZA5     LDX  I1 IDSA1       SET POINTER AT STMNT ID
288          SLA     16
289          STO  L  SW3         CLEAR SWITCH 3
290          STO  L  SW4         CLEAR SWITCH 4
291          STO  L  SW6         CLEAR SWITCH 6
292          LD   L  NCNTS       RESET NORM TO
293          STO  L  NCNT        ORIGINAL VALUE
294          BSI  L  GETID       GET ID TYPE
295          LD      IDWD        LOAD ID WORD
296          S    L  ID48+2      IS DATA STATEMENT
297          BSC  L  *+2,Z       BRANCH IF NO
298          STX  L0 SW7         SET DATA STMNT INDICATOR
299          LD      IDWD        LOAD ID WORD
```

```
300          S    L  ID7+2       IS END STATEMENT
301          BSC  L  ZA1A,+-     BRANCH IF YES
302  *                       PUT ID WORD ON THE NEW STRING
303  ZA6     STX  L2 IDSV2       STORE OUTPUT POINTER
304          LDX  I3 IDSV1       LOAD ADDR OF STMNT ID WD
305          LD    3 0
306          OR       IDWD
307          BSI  L  PUT         PLACE ID WORD ON STRING
308          LD    3 0           LOAD ID WORD
309          BSC      E          HAS STMNT NO.
310          MDX      *+1        YES, GO PUT NO. ON STRING
311          MDX      ZA6A       NO, GO SCAN BODY OF STMNT
312          LD    3 1           LOAD 1ST WORD OF STMNT NO
313          BSI  L  PUT         PUT ON OUTPUT STRING
314          LD    3 2           LOAD 2ND WORD OF STMNT NO
315          BSI  L  PUT         PUT ON OUTPUT STRING
316  *                       INITIALIZE TO SCAN BODY OF STAT
317  ZA6A    SLA      16         CLEAR SWITCHES
318          STO      SW1
319          STO      SW2
320          STO      FCNT
321  *                       IS IDWD REAL OR INTEGER
322          LD       IDWD       LOAD ID WORD
323          S        REALC      IS IT REAL
324          BSC  L   *+3,+-     BRANCH IF YES
325          S        INTC       IS IT INTEGER
326          BSC  L   ZA7,Z      BRANCH IF NOT
327  *                       TAG SW1
328          STX    0 SW1        SET REAL/INTEGER INDICATO
329          MDX      ZA8        GO GET NEXT CHARACTER
330  ZA7     S        DOC        IS IT DO STMNT
331          BSC  L   ZA8,Z      BRANCH IF NOT
332  *                       TAG SW2
333          STX    0 SW2        SET DO INDICATOR
334  ZA8     BSI  L   GET1       GET CHAR FROM INPUT STRIN
335          STO      X          STORE CHAR
336          LD       SW1        IS IT REAL/INTEGER STMNT
337          BSC  L   ZA11,+-    BRANCH IF NOT
338          LD       FCNT       LOAD FUNCTION LETTER CTR
339          S        EIGHT      ALL LETTERS COMPARED
340          BSC  L   ZA9,+-     YES, GO PUT ON ISOLATOR
341          MDX  L   FCNT,1     MODIFY FUNCTION LETTER CT
342  *                       ISOLATES THE WORD FUNCTION IN
343  *                       A TYPE STMNT ON THE STRING
344          LDX  L3  *-*        LOAD XR3 WITH
345  FCNT    EQU      *-1        FUNCTION LETTER COUNTER
346          LD   L3  FUNT-1     LOAD LETTER FROM TABLE
347          S        X          DOES IT MATCH STRING CHAR
348          BSC  L   ZA12,+-    BRANCH IF YES
349          MDX      ZA10       GO NORMALIZE SWITCH 1
350  ZA9     LD       COLON      LOAD COLON
351          BSI  L   MAKE       CONVERT CHAR FOR OUTPUT
352  *                       NORMALIZE SW1
353  ZA10    SLA      16
354          STO      SW1        CLEAR REAL/INTEGER SWITCH
355          MDX      ZA12       GO TEST FOR LAST CHAR
356  *                       ISOLATES STMNT NUMBERS IN
357  *                       DO STATEMENTS
358  ZA11    LD       SW2        IS IT DO STMNT
359          BSC  L   ZA12,+-    BRANCH IF NOT
```

```
*                               IS X NUMERIC
         LD          X          LOAD CHARACTER
         SLA         11         SHIFT LEFT 11
         BSC    L    *+2,-      FIRST TEST FOR NUMERIC
         BSC    L    ZA12,C     SECOND TEST-BRANCH NUMERI
         LD          COLON      LOAD COLON
         BSI    L    MAKE       CONVERT CHAR FOR OUTPUT
*                               NORMALIZE SW2
         SLA         16
         STO    L    SW2        CLEAR DU INDICATOR
ZA12     LD     L    X          LOAD CHARACTER
         BSC    L    ZA13,+-    BRANCH IF FINISHED STMNT
         LD     L    X          LOAD CHARACTER
         BSI    L    MAKE       CONVERT CHAR FOR OUTPUT
         MDX         ZA8        CONTINUE CHECK LOOP
*                               CONSTANTS
SW3      DC          0          SWITCH 3
SW4      DC          0          SWITCH 4
FUNT     DC          /C6        F
         DC          /E4        U
         DC          /D5        N
         DC          /C3        C
         DC          /E3        T
         DC          /C9        I
         DC          /D6        O
         DC          /D5        N
COLON    DC          /3A        COLON
QSX1     DC          0          TEMPORARY STORAGE
SF803    DC          /F803      CLEAR NORM MASK
SFF      DC          /FF        CLEAR LEFT 8 BITS MASK
SEMI     DC          /1E        SEMI-COLON
*                               PLACES THE STMNT TERMINATOR
*                               (SEMI-COLON) AT THE END OF THE
*                               STATEMENT
ZA13     LD          SEMI       LOAD SEMI-COLON
         BSI    L    MAKE       CONVERT CHAR FOR OUTPUT
         LD     L    IDWD       LOAD ID WORD
         S      L    ID4+4      IS CONTINUE STMNT
         BSC    L    QZA1,Z     BRANCH IF NO
         MDX       2 -1         DECREMENT OUTPUT POINTER
*                               INSERT THE STATEMENT NORM
*                               STMNT ID WORD
QZA1     STX       2 QSX1       STORE OUTPUT POINTER
         LD          QSX1
         S           IDSV2      SUBTRACT START ADDRESS TO
         SLA         2          GET NORM - MOVE IN ID WOR
         STO         QSX1
         LD     L    *-*        LOAD STATEMENT
IDSV2    EQU         *-1        ID WORD
         AND         SF803      ZERO NORM
         OR          QSX1       INSERT NEW NORM
         STO    I    IDSV2      PUT BACK ON STRING
         BSC    L    MOVE1      GO TO NEXT STATEMENT
*                               GETS A CHAR FROM THE I/P STRING
GET1     DC          0          RETURN ADDRESS
         LD     L    SW3        IS CHAR IN RIGHT HALF
         BSC    L    GXY1,Z     BRANCH IF NOT
         LD     L    SW6        IS IT END OF STMNT
         BSC    L    GXY4,Z     BRANCH IF NOT
*                               TAG SW3
```

```
420           STX   L0 SW3          SET RIGHT HALF INDICATOR
421   *                        GET LEFT CHARACTER
422           LD     1 0           LOAD WORD FROM STRING
423           SRA      8           UNPACK - GET LEFT HALF
424           MDX      GXY3        GO TO RETURN
425   GXY1    SLA      16
426           STO   L  SW3         CLEAR RIGHT HALF INDICATO
427   *                        GET RIGHT CHARACTER
428           LD     1 0           LOAD INPUT WORD
429           AND   L  SFF         UNPACK - GET RIGHT CHAR
430   GXY2    MDX    1 1           MOVE POINTER
431           MDX   L  NCNT,-1     DECREMENT NORM COUNTER
432           MDX      GXY3        BRANCH TO RETURN
433           STX    0 SW6         SET END OF STMNT SW
434   *                        BLANK CHARACTER
435   GXY5    S        S0040       IS LAST CHAR BLANK
436           BSC   L  GXY3,+-     BRANCH IF YES
437           A        S0040       SET LAST CHAR TO BLANK
438           MDX      GXY3        BRANCH TO RETURN
439   GXY4    SLA      16          SET END OF STMNT INDICATO
440   GXY3    BSC   I  GET1        RETURN
441   *                        GETS TWO CHARACTERS FROM THE
442   *                        INPUT STRING
443   GET2    DC       0           RETURN ADDRESS
444           LD       SW6         IS IT END OF STMNT
445           BSC   L  GXY4,Z      BRANCH IF YES
446   *                        GET BOTH CHARACTERS
447           LD       GET2        LOAD RETURN ADDRESS
448           STO      GET1        STORE IN GET1
449           LD     1 0           LOAD WORD IN STMNT
450           MDX      GXY2        GO TO MOVE POINTER
451   *                        CONSTANTS
452   NCNT    DC       0           NORM COUNTER
453   NCNTS   DC       0           SAVE ORIGINAL NORM
454   SW6     DC       0           SWITCH 6
455   ERID    DC       /A008       ERROR STMNT ID WORD
456   NINE    DC       9           NINE
457   CHAR    DC       0           CHARACTER COUNTER
458   ERNO    DC       4           ERROR NO, 4
459   OVLPC   DC       0           OVERLAY TEST WORD
460   S0040   DC       /0040       EBC BLANK
461   GETID   DC       0           RETURN ADDRESS
462           LDX   L3 TWONT-2     INITIALIZE TABLE POINTER
463   *                        INITIALIZES FOR A STATEMENT
464   *                        TYPE CODE TABLE SEARCH
465   *                        GETS THE STMNT TYPE CODE
466   *                        STORES IT IN THE ID WORD
467   GTID1   MDX    3 2           MODIFY POINTER
468           LD     3 0           LOAD WORD FROM TABLE
469           BSC   L  ER,+-       BRANCH IF END OF TABLE
470           S      1 0           DOES MATCH WORD IN STRING
471           BSC   L  GTID1,Z     CONTINUE LOOP IF NOT
472   *                        GET CORRECT NAME TABLE
473           LD     3 1           LOAD ADDR OF NAME TABLE
474           STO      *+2
475           STX    3 GTID4+3     SAVE TABLE POINTER
476           LDX   L3 *-*         ADDR OF NAME TABLE
477   *                        INITIALIZE TO SCAN NAME TABLE
478           STX    1 GTID4+1     SAVE INPUT POINTER
479           MDX    1 1           MOVE INPUT POINTER
```

```
480  *                          PUT CHARACTER COUNT IN CHAR
481          LD     3 0          LOAD CHARACTER COUNT
482          STO      CHAR       SAVE
483          BSC  L   GTID3,+-   BRANCH IF ZERO
484  GTID2   MDX  L   CHAR,-1    DECREMENT COUNT
485          MDX      GTID6      GET NEXT WORD FROM STRING
486          BSI      GET1       GET CHAR FROM INPUT STRIN
487          MDX      GTID5      SEE IF CHARACTERS MATCH
488  GTID6   BSI      GET2       GET NEXT WORD IN STMNT
489  GTID5   S      3 1          DOES WORD MATCH
490          BSC  L   GTID4,Z    BRANCH IF NO
491          MDX    3 1          MOVE TABLE POINTER
492          MDX  L   CHAR,-1    DECREMENT CHARACTER COUNT
493          MDX      GTID2      GO TO DECREMENT COUNT
494  GTID3   LD     3 1          LOAD STMNT ID WORD
495          STO  L   IDWD       SAVE
496          MDX  L   NCNT,-1    MODIFY NORM COUNT
497          MDX      *+1        SKIP SET SWITCH
498          STX    0 SW6        SET END OF STMNT SWITCH
499          BSC  I   GETID      RETURN
500  GTID4   LDX  L1  *-*        RESTORE INDEX 1
501          LDX  L3  *-*        RESTORE INDEX 3
502          LD       NCNTS      RESET NORM TO
503          STO      NCNT       ORIGINAL VALUE
504          SLA      16
505          STO  L   SW3        CLEAR RIGHT HALF INDICATO
506          MDX      GTID1      START COMPARE AGAIN
507  *                THIS SUBROUTINE PUTS THE WORD THAT
508  *                IS IN THE ACCUMULATOR ON THE
509  *                OUTPUT STRING, ENTRY IS A BSI
510  *                INSTRUCTION TO THE LABEL (PUT),
511  *                THE OUTPUT STRING POINTER (XR2) IS
512  *                INCREMENTED BY ONE UPON EXITING,
513  *                     PUT WORD ON STRING
514  PUT     DC       0          RETURN ADDRESS
515          STO    2 0          PUT WORD ON OUTPUT STRING
516          MDX    2 1          MOVE OUTPUT STRING POINTE
517  *                     OVERLAP ERROR
518          STX    2 OVLPC      STORE OUTPUT POINTER
519          LD       OVLPC
520          S        THREE      SUBTRACT THREE
521          S    L   EOFST      SUBTRACT END OF SYMBOL TB
522          BSC  L   RTN,+Z     BRANCH IF NO OVERLAP
523  *                     SET UP OVERLAP ERROR
524          MDX  L   ERROR,1    SET OVERLAP ERROR
525          BSC  L   WAIT       EXIT FROM PHASE
526  RTN     BSC  I   PUT        RETURN
527  *                     PUT OUT ERROR NO, 4
528  ER      LD       ERID       LOAD ERROR STMNT ID WORD
529          BSI      PUT        PUT ON OUTPUT STRING
530          LD   I   IDSV1      LOAD STMNT ID WORD
531          BSC  L   ER1,E      BRANCH IF HAVE STMNT NO,
532  ER2     LD       ERNO       LOAD ERROR NUMBER
533          BSI      PUT        PUT ON OUTPUT STRING
534          BSC  L   MOVE1      GO TO NEXT STATEMENT
535  ER1     LD     2 -1         LOAD OUTPUT STMNT ID WORD
536          A        NINE       ADD TWO TO NORM, SET STMN
537          STO    2 -1         NO, BIT ON, O/P ON STRING
538          LDX  I3  IDSV1      LOAD STMNT ADDR IN XR3
539          LD     3 1          LOAD FIRST WORD OF NUMBER
```

```
540         BSI      PUT         PUT ON OUTPUT STRING
541         LD     3 2           LOAD 2ND WORD OF STMNT NO
542         BSI      PUT         PUT ON OUTPUT STRING
543         MDX      ER2         PUT ERROR NO, ON STRING
544   *                      CONSTANTS
545   THREE DC       3           THREE
546   SVV1  DC       0           SAVE WORD
547   SW5   DC       0           SWITCH 5
548   S0001 DC       /0001       ONE
549   S003F DC       /003F       NORM MASK
550   QUOTE DC       /7D         QUOTE
551         BSS  E   0
552   NAMS  DC       0           TWO WORD AREA FO
553         DC       0           SYMBOL TABLE NAME
554   *                      STORES NAME IN NAME CODE
555   *                      STORES OPTR IN ONE WORD ON
556   *                      THE STRING
557   MAKE  DC       0           RETURN ADDRESS
558         STO      SVV1        STORE WORD
559         LD   L   SW8         HAS QUOTE BEEN ENCOUNTERE
560         BSC  L   MAKE7,Z     BRANCH IF YES
561         LD       SVV1        LOAD WORD
562         SLA      8           GET RIGHT CHAR
563         BSC  L   MAKE1,-     BRANCH IF SPECIAL OPERATO
564         SLA      16
565         STO      SW5         CLEAR ARGUMENT TYPE SW
566         LD   L   SW4         IS THERE CHAR IN NAMS
567         BSC  L   MAKE4,Z     BRANCH IF YES
568   *                      INITIALIZE NAMS
569   MAKE3 SLT      16          CLEAR EXTENSION
570         LD       S0001       LOAD NAMS FULL INDICATOR
571         STD      NAMS        STORE IN NAMS
572         STX  LO  SW4         SET CHAR IN NAMS SW
573   *                      IS NAMS FULL
574   MAKE4 LD       NAMS+1      LOAD 2ND WORD OF NAMS
575         SLT      1           SHIFT TO SEE IF FULL
576         BSC  L   MAKE2,+Z    BRANCH IF FULL
577   *                      ADD ARGUMENT TO NAMS
578         LDD      NAMS        LOAD NAME
579         RTE      26          CLEAR SPACE FOR NEXT CHAR
580         STO      NAMS        STORE
581         LD       SVV1        LOAD WORD
582         AND      S003F       CLEAR EXTRA BITS
583         OR       NAMS        COMBINE WITH NAMS
584         STD      NAMS        SAVE NEW NAMS
585   MAKE5 BSC  I   MAKE        RETURN
586   MAKE1 STX    0 SW5         SET SWITCH TO OPERATOR
587         LD   L   SW4         IS THERE CHAR IN NAMS
588         BSC  L   MAKE7,+-    BRANCH IF NO
589   *                      NORMALIZE NAMS
590   MAKE2 LDD      NAMS        LOAD NAMS
591         RTE      16          ROTATE NAME
592         SLT      1           MOVE CHAR FROM EXTENSION
593   *                           TO ACCUMULATOR
594         BSC  L   *-3,-       BRANCH IF SPECIAL OPERATO
595         STD      NAMS        STORE NAME
596         BSI      PUT         PUT FIRST WORD ON STRING
597         LDD      NAMS        LOAD NAME
598         SRT      3           PUT FIRST HALF OF 3RD CHA
599         SLA      16          INTO EXTENSION-MOVE ENTIR
```

```
600          SLT     6          CHAR TO ACC FOR TESTING
601          BSC  L  *+4,+-     BRANCH IF ONE WORD NAME
602          LDD     NAMS       LOAD NAMS
603          SRA     15         MOVE BIT TO RIGHT-MOST PO
604          SLT     15         MOVE EXTENSION TO ACC
605          BSI     PUT        PUT ON OUTPUT STRING
606  *                      GET MODE OF NAMS
607          LDX  L3 /8C00      LOAD FLOATING POINT MODE
608          LD   L  CCWD       TEST EXTENDED PRECISION
609          SLA     13         BIT OF CONTROL CARD WORD
610          BSC  L  *+2,-      BRANCH IF  STANDARD PREC
611          LDX  L3 /8A00      LOAD EXTENSION PRECISION
612  *                           MODE
613          STX   3 NAMED      STORE MODE
614  *                    FIND NAME TO TRANSLATE
615          LDX   3 36         INITIALIZE COUNTER
616  EXFN1   LD    2 -2         LOAD WORD TO BE TRANSLATE
617          S    L3 FUNEX-4    MATCH ENTRY IN TABLE
618          BSC  L  EXFN2,Z    BRANCH IF NOT
619          LD    2 -1         LOAD 2ND WORD
620          S    L3 FUNEX-3    MATCH ENTRY IN TABLE
621          BSC  L  EXFN2,Z    BRANCH IF NOT
622          LDD  L3 FUNEX-2    LOAD FUNCTION NAME
623          OR      NAMED      ADD MODE TO NAME
624          STO   2 -2         STORE IN OUTPUT STRING
625          RTE     16         LOAD 2ND WORD
626          STO   2 -1         PUT ON OUTPUT STRING
627          MDX     *+2        GO TEST SWITCH 5
628  EXFN2   MDX   3 -4         MODIFY COUNTER
629          MDX     EXFN1      CONTINUE LOOP
630          LD      SW5        IS ARGUMENT OPERATOR
631          BSC  L  MAKE3,+-   BRANCH IF NO
632  MAKE7   LD      SVV1       LOAD WORD
633          S       QUOTE      IS IT QUOTE
634          BSC  L  *+3,+-     SKIP AROUND IF YES
635          S       QUMSL      ELSE COMPARE TO SLASH
636          BSC  L  MAKE9,Z    BR IF NO COMPARE
637          LD   L  SW7        IS IT DATA STMNT
638          BSC  L  MAKE6,+-   BRANCH IF NO
639          SLA     16
640          STO  L  SW9        SET LEFT HALF SWITCH
641          LD      *
642          EOR  L  SW8        REVERSE QUOTE INDICATOR
643          STO  L  SW8
644          MDX     MAKE6      PUT CHAR ON OUTPUT STRING
645  MAKE9   LD   L  SW8        WAS QUOTE ENCOUNTERED
646          BSC  L  MAKE6,+-   BRANCH IF NOT
647          LD   L  SW9        OUTPUT RIGHT HALF OF WORD
648          BSC  L  MAKE8,Z    BRANCH IF YES
649          LD      SVV1       LOAD WORD
650          SLA     8          SHIFT TO LEFT
651          BSI  L  PUT        PUT ON OUTPUT STRING
652          STX  L0 SW9        SET SW TO O/P RIGHT HALF
653          MDX     MAKEA      GO NORMALIZE SWITCH 4
654  MAKE8   LD    2 -1         LOAD WORD FROM OUTPUT
655          OR   L  SVV1       PUT IN RIGHT CHAR
656          STO   2 -1         PUT BACK ON OUTPUT STRING
657          SLA     16
658          STO  L  SW9        SET SW TO O/P LEFT HALF
659          MDX     MAKEA      GO NORMALIZE SWITCH 4
```

```
660  MAKE6 LD    L   SVV1        LOAD WORD
661        AND   L   S003F       GET NECESSARY BITS
662        BSI   L   PUT         PUT ON OUTPUT STRING
663  MAKEA SLA       16
664        STO   L   SW4         CLEAR CHAR IN NAMS SW
665        MDX       MAKE5       RETURN
666  QUMSL DC        /1E-/7D     QUOTE - SEMICOLON
667  *                     FUNCTIONAL EXCHANGE TABLE
668        BSS   E   0
669  FUNEX DC        /C44A       -SI-
670        DC        /D000       -N -
671        DC        /0111       - S-
672        DC        /9540       -IN-
673        DC        /8684       -CO-
674        DC        /A000       -S -
675        DC        /001A       - C-
676        DC        /E880       -OS-
677        DC        /8318       -AT-
678        DC        /9540       -AN-
679        DC        /000C       -AT-
680        DC        /8055       -AN-
681        DC        /C4C3       -SQ-
682        DC        /98C0       -RT-
683        DC        /0113       -SQ-
684        DC        /8663       -RT-
685        DC        /829A       -AL-
686        DC        /E1C0       -OG-
687        DC        /000A       -AL-
688        DC        /8587       -OG-
689        DC        /8B3A       -EX-
690        DC        /F000       -D -
691        DC        /002C       - E-
692        DC        /F5C0       -XP-
693        DC        /C448       -SI-
694        DC        /F540       -GN-
695        DC        /0111       -SI-
696        DC        /91D5       -GN-
697        DC        /C60A       -TA-
698        DC        /D200       -NH-
699        DC        /0118       -TA-
700        DC        /9548       -NH-
701        DC        /8214       -AB-
702        DC        /A000       -S -
703        DC        /0008       - A-
704        DC        /A880       -BS-
705  NAMED DC        0           USEFUL CONSTANT
706  *                     TABLE CONTAINING THE FIRST TWO
707  *                     CHARS OF THE FORTRAN STMNT
708  *                     NAMES AND THE ADDRESSES OF THE
709  *                     REMAINING NAME CHARACTERS
710  TWONT DC        /C4D6       DO
711        DC        ID6         ADDRESS
712        DC        /C7D6       GO TO
713        DC        ID11        ADDRESS
714        DC        /C9C6       IF
715        DC        ID12        ADDRESS
716        DC        /D9C5       READ
717        DC        ID15        ADDRESS
718        DC        /E6D9       WRITE
719        DC        ID21        ADDRESS
```

```
720          DC       /C3C1      CALL EXIT
721          DC       ID1J       ADDRESS
722          DC       /C3C1      CALL LINK
723          DC       ID1K       ADDRESS
724          DC       /C3C1      CALL
725          DC       ID2        ADDRESS
726          DC       /C6C9      FIND
727          DC       ID9S       ADDRESS
728          DC       /C4C9      DIMENSION
729          DC       ID5        ADDRESS
730          DC       /C3D6      COMMON
731          DC       ID3        ADDRESS
732          DC       /C5D8      EQUIVALENCE
733          DC       ID9        ADDRESS
734          DC       /C9D5      INTEGER
735          DC       ID13       ADDRESS
736          DC       /D9C5      REAL
737          DC       ID16       ADDRESS
738          DC       /C5E7      EXTERNAL
739          DC       ID9J       ADDRESS
740          DC       /C3D6      CONTINUE
741          DC       ID4        ADDRESS
742          DC       /D7C1      PAUSE
743          DC       ID14       ADDRESS
744          DC       /D9C5      RETURN
745          DC       ID17       ADDRESS
746          DC       /E2E3      STOP
747          DC       ID19       ADDRESS
748          DC       /D9C5      REWIND
749          DC       ID18       ADDRESS
750          DC       /C2C1      BACKSPACE
751          DC       ID1        ADDRESS
752          DC       /C5D5      END FILE
753          DC       ID8        ADDRESS
754          DC       /C4C5      DEFINE FILE
755          DC       ID4J       ADDRESS
756          DC       /C5D5      END
757          DC       ID7        ADDRESS
758          DC       /C6E4      FUNCTION
759          DC       ID10       ADDRESS
760          DC       /E2E4      SUBROUTINE
761          DC       ID20       ADDRESS
762          DC       /C4C1      DATA
763          DC       ID4B       ADDRESS
764          DC       0          USEFUL CONSTANT
765  *                        BODY NAME TABLE
766  *                        BACKSPACE
767  ID1     DC       7          COUNT
768          DC       /C3D2      CK
769          DC       /E2D7      SP
770          DC       /C1C3      AC
771          DC       /00C5      E
772          DC       /0800      ID WORD
773  *                        CALL EXIT
774  ID1J    DC       6          COUNT
775          DC       /D3D3      LL
776          DC       /C5E7      EX
777          DC       /C9E3      IT
778          DC       /E002      ID WORD
779  *                        CALL LINK
```

```
780  ID1K   DC      6          COUNT
781         DC      /D3D3      LL
782         DC      /D3C9      LI
783         DC      /D5D2      NK
784         DC      /E000      ID WORD
785  *                    CALL
786  ID2    DC      2          COUNT
787         DC      /D3D3      LL
788         DC      /3000      ID WORD
789  *                    COMMON
790  ID3    DC      4          COUNT
791         DC      /D4D4      MM
792         DC      /D6D5      ON
793         DC      /3800      ID WORD
794  *                    CONTINUE
795  ID4    DC      6          COUNT
796         DC      /D5E3      NT
797         DC      /C9D5      IN
798         DC      /E4C5      UE
799         DC      /8000      ID WORD
800  *                    DATA
801  ID4B   DC      2          COUNT
802         DC      /E3C1      TA
803         DC      /F800      ID WORD
804  *                DEFINE FILE
805  ID4J   DC      8          COUNT
806         DC      /C6C9      FI
807         DC      /D5C5      NE
808         DC      /C6C9      FI
809         DC      /D3C5      LE
810         DC      /F000      ID WORD
811  *                    DIMENSION
812  ID5    DC      7          COUNT
813         DC      /D4C5      ME
814         DC      /D5E2      NS
815         DC      /C9D6      IO
816         DC      /00D5      N
817         DC      /4000      ID WORD
818  *                    DO
819  ID6    DC      0          COUNT
820         DC      /5800      ID WORD
821  *                    END
822  ID7    DC      2          COUNT
823         DC      /C440      D
824         DC      /1000      ID WORD
825  *                    ENDFILE
826  ID8    DC      5          COUNT
827         DC      /C4C6      DF
828         DC      /C9D3      IL
829         DC      /00C5      E
830         DC      /1800      ID WORD
831  *                    EQUIVALENCE
832  ID9    DC      9          COUNT
833         DC      /E4C9      UI
834         DC      /E5C1      VA
835         DC      /D3C5      LE
836         DC      /D5C3      NC
837         DC      /00C5      E
838         DC      /A800      ID WORD
839  *                EXTERNAL
```

```
840  ID9J  DC      6          COUNT
841        DC      /E3C5      TE
842        DC      /D9D5      RN
843        DC      /C1D3      AL
844        DC      /C800      ID WORD
845  *                  FIND
846  ID9S  DC      2          COUNT
847        DC      /D5C4      ND
848        DC      /E800      ID WORD
849  *                  FUNCTION
850  ID10  DC      6          COUNT
851        DC      /D5C3      NC
852        DC      /E3C9      TI
853        DC      /D6D5      ON
854        DC      /6800      ID WORD
855  *                  GOTO
856  ID11  DC      2          COUNT
857        DC      /E3D6      TO
858        DC      /7000      ID WORD
859  *                  IF
860  ID12  DC      0          COUNT
861        DC      /7800      ID WORD
862  *                  INTEGER
863  ID13  DC      5          COUNT
864        DC      /E3C5      TE
865        DC      /C7C5      GE
866        DC      /00D9      R
867        DC      /5000      ID WORD
868  *                  PAUSE
869  ID14  DC      3          COUNT
870        DC      /E4E2      US
871        DC      /00C5      E
872        DC      /9800      ID WORD
873  *                  READ
874  ID15  DC      2          COUNT
875        DC      /C1C4      AD
876        DC      /9000      ID WORD
877  *                  REAL
878  ID16  DC      2          COUNT
879        DC      /C1D3      AL
880        DC      /4800      ID WORD
881  *                  RETURN
882  ID17  DC      4          COUNT
883        DC      /E3E4      TU
884        DC      /D9D5      RN
885        DC      /8000      ID WORD
886  *                  REWIND
887  ID18  DC      4          COUNT
888        DC      /E6C9      WI
889        DC      /D5C4      ND
890        DC      /2800      ID WORD
891  *                  STOP
892  ID19  DC      2          COUNT
893        DC      /D6D7      OP
894        DC      /B800      ID WORD
895  *                  SUBROUTINE
896  ID20  DC      8          COUNT
897        DC      /C2D9      BR
898        DC      /D6E4      OU
899        DC      /E3C9      TI
```

```
900         DC      /D5C5       NE
901         DC      /2000       ID WORD
902   *                   WRITE
903   ID21  DC      3           COUNT
904         DC      /C9E3       IT
905         DC      /00C5       E
906         DC      /8800       ID WORD
907   *                   RE-INITIALIZE
908   CALLC LDX   I1 SOFS       INITIALIZE INPUT POINTER
909   *                   CHECKS FOR THE PRESENCE OF AN
910   *                   END STATEMENT
911   ENDD  STX     1 IDSVV     STORE INPUT POINTER
912         LD      1 0         LOAD STMNT ID WORD
913         S     L   S1000     IS IT END STMNT
914         BSC   L   WAIT,+-   BRANCH IF YES
915   *                   CALL STATEMENT
916         LD      1 0         LOAD STMNT ID WORD
917         AND       IDTPE     GET ID TYPE
918         SRA       2         SHIFT RIGHT TWO
919         S         CALL1     IS IT CALL STMNT
920         BSC   L   MOVIE,Z   BRANCH IF NOT
921         LD      1 0         LOAD STMNT ID WORD
922         BSC       E         HAS STMNT NO.
923         MDX     1 2         YES - MOVE PT OVER NO.
924         MDX     1 1         MOVE INPUT POINTER
925   *                   CHECKS FOR THE NAME SSWTCH
926   IDAHO LD      1 0         LOAD FIRST WORD IN BODY
927         S         SENC      IS IT SS
928         BSC   L   XYZ,Z     BRANCH IF NOT
929         LD      1 1         LOAD 2ND WORD OF BODY
930         S         SENC1     IS IT WT
931         BSC       Z         SKIP IF YES
932         MDX       XYZ       GO TO NEXT COMPARE
933         LD      1 2         LOAD 3RD WORD OF BODY
934         S         SENC2     IS IT CH
935         BSC       Z         SKIP IF YES
936         MDX       XYZ       GO TO NEXT COMPARE
937         MDX       XXYZ      GO TO CONVERT TO 5 CHAR
938   *                   CONSTANTS
939   IDSVV DC        0         INPUT POINTER
940   IDTPE DC        /F800     ID MASK
941   CALL1 DC        /0C00     CALL STMNT ID TYPE
942   SENC  DC        /C514     SS
943   SENC1 DC        /E8C3     WT
944   SENC2 DC        /9000     CH
945   NAME  DC        0         INPUT POINTER
946   OVER  DC        /AD28     OV
947   OVER1 DC        /D646     ER
948   OVER2 DC        /A600     FL
949   SENL  DC        /C499     SL
950   SENL1 DC        /98C5     IT
951   SENL2 DC        /C600     ET
952   SENL3 DC        /98E3     T
953   S0004 DC        4         NORM OF ONE
954   *                   CHECKS FOR THE NAME OVERFL
955   XYZ   LD      1 0         LOAD 1ST WORD IN BODY
956         S         OVER      IS IT OV
957         BSC   L   XYZ1,Z    BRANCH IF NOT
958         LD      1 1         LOAD 2ND WORD IN BODY
959         S         OVER1     IS IT ER
```

```
 960          BSC  L  XYZ1,Z     BRANCH IF NOT
 961          LD    1 2          LOAD 3RD WORD IN BODY
 962          S       OVER2      IS IT FL
 963          BSC  L  XXYZ,+-    BRANCH IF YES
 964  *                       CHECKS FOR THE NAME SLITET
 965  XYZ1    LD    1 0          LOAD 1ST WORD IN BODY
 966          S       SENL       IS IT SL
 967          BSC  L  MOVIE,Z    BRANCH IF NOT
 968          LD    1 1          LOAD 2ND WORD IN BODY
 969          S       SENL1      IS IT IT
 970          BSC     Z          SKIP IF YES
 971          MDX     MOVIE      GO TO NEXT STMNT
 972          LD    1 2          LOAD 3RD WORD IN BODY
 973          S       SENL2      IS IT ET
 974          BSC     Z          SKIP IF YES
 975          MDX     MOVIE      GO TO NEXT STMNT
 976          LD      SENL3      LOAD NEW WORD FOR SLITET
 977          STO   1 1          STORE END
 978  *                       CLOSES THE STRING BY ONE WORD
 979  *                       AND ADJUSTS THE STMNT NORM
 980  XXYZ    MDX   1 2          MOVE INPUT POINTER
 981          STX   1 NAME       SAVE INPUT POINTER
 982          LD   L  EOFS       LOAD END OF STRING ADDRES
 983          S       NAME       SUBTRACT INPUT POINTER TO
 984          STO     *+1        GET RANGE OF LOOP
 985          LDX   L3 0         LOAD XR3 WITH RANGE
 986          LD    1 1          MOVE WORD DOWN
 987          STO   1 0          ONE POSITION
 988          MDX   1 1          MOVE POINTER
 989          MDX   3 -1         DECREMENT RANGE
 990          MDX     *-5        CONTINUE LOOP
 991  *                       ADJUST NORM
 992          LDX   I1 IDSVV     RESTORE INPUT POINTER
 993          LD    1 0          LOAD STMNT ID WORD
 994          S       S0004      DECREMENT NORM BY ONE
 995          STO   1 0          PUT NORM BACK ON STRING
 996  *                       UPDATES THE STRING I/P PT (XR1)
 997  *                       TO MOVE TO THE NEXT STMNT
 998  MOVIE   LDX   I1 IDSVV     RESTORE INPUT POINTER
 999          LD    1 0          LOAD STMNT ID WORD
1000          SLA     5          SHIFT TO GET NORM
1001          SRA     7
1002          A       IDSVV      ADD INPUT POINTER TO GET
1003          STO     NXID+1     ADDRESS OF NEXT STMNT
1004  NXID    LDX   L1 0         RESET INPUT POINTER
1005          MDX     ENDD       GO CHECK FOR STMNT TYPE
1006  *               PROCESS IOCS WORD AND SET DEVICE
1007  *               INITIALIZATION ROUTINES
1008  *               SETUP PRECISION AND TEST FOR ONE WD
1009  *               INTEGERS
1010  CQCT    SLA     16         ZERO THE ACC
1011          LD   L  CCWD       LD CONTROL CARD WORD
1012          SLA     7          TEST FOR NONPROCESS PROG
1013          BSC  L  *+5,+Z     BRANCH IF BIT 7 ON
1014          LD   L  CCWD       LD CCWD IF PROCESS PROG
1015          OR      Z1         SET BIT 9 ON FOR ONE WD IN
1016          STO  L  CCWD       STORE THE CCWD IN CCWD
1017          BSC  L  ZAO        GO TO SCAN STMNT STRING
1018  Z1      DC      /0040      BIT 9 ON FOR ORING
1019          BSS     OVERL-*+320*3
```

```
1020          END     START
```