SEQTR 1

```
*   GENERAL AUTOMATION, INC,  ALL RIGHTS RESERVED
******************************************************
*
*   PROGRAM NAME  FORTRAN PH-03
*
*   MODEL NUMBER  8F003
*
*   PURPOSE        FORTRAN PHASE-03
*
*   PROGRAMMER     DICK WALLMANN, MODS-MARK ELFIELD
*
******************    REVISION LIST    **************
*
*   RV DATE      SCO    BY  REASON FOR CHANGE
*   -- ---- ---- ----- --- ----------------------
*
*   01 11/16/70         RPH INITIAL RELEASE
*
******************************************************
******************************************************
* GA 18/30  FORTRAN COMPILER    05/ 1/70
*STATUS -  VERSION 1, MODIFICATION 0
*FUNCTION/ OPERATION-
*     * CHECKS SUBPROGRAM AND SPECIFICATION STMNTS
*       FOR THE PROPER ORDER. REMOVES ANY STMNT
*       NUMBERS FROM THESE STATEMENTS
*     * CHECKS TO ENSURE THAT STMNTS FOLLOWING IF,
*       GO TO, RETURN, AND STOP STMNTS HAVE STMNT
*       NUMBERS
*     * REMOVES CONTINUE STATEMENTS THAT DO NOT HAVE
*       STATEMENT NUMBERS
*     * CHECKS THE STMNTS FOR STMNT NUMBERS
*     * CHECKS THE SYMBOL TABLE FOR A PREVIOUS ENTRY
*       OF THE SAME STMNT NUMBER
*     * PLACES THE STMNT NO, INTO THE SYMBOL TABLE
*     * PLACES THE SYMBOL TABLE ADDRESS INTO THE
*       STRING ENTRY
*     * PHASE 3 MAKES TWO PASSES THROUGH THE STMNT
*       STRING. THE FIRST PASS CHECKS TO ASCERTAIN
*       THAT THE SUBPROGRAM AND SPECIFICATION
*       STMNTS ARE IN THE FOLLOWING SEQUENCE
*          SUBROUTINE OR FUNCTION STATEMENT
*          EXTERNAL STATEMENTS
*          DIMENSION STATEMENTS
*          COMMON STATEMENTS
*          EQUIVALENCE STATEMENTS
*     * THE SECOND PASS OF PHASE 3 SCANS THE STMNT
*       STRING FOR STMNTS WITH STMNT NUMBERS
*ENTRY POINTS-
*     * START IS THE ENTRY ADDRESS LABEL.  PHASE 3
*       IS LOADED BY PHASE 2 VIA A CALL FROM ROLRX.
*INPUT-
*   NONE IN THE USUAL SENSE, HOWEVER, THE STMNT
*   STRING SCANNED BY THE COMPILER MAY BE
*   CONSIDERED INPUT
*OUTPUT-
*   NONE IN THE USUAL SENSE, HOWEVER, PHASE 3
*   REMOVES STMNTS FROM THE STMNT STRING AND ADDS
*   ENTRIES TO THE SYMBOL TABLE
```

```
*EXTERNAL ROUTINES-N/A
*EXITS-
*    * NORMAL - PHASE 3 EXITS NORMALLY IF NO OVER-
*               LAP ERROR HAS OCCURRED.
*    * ERRORS - IF THERE IS AN OVERLAP OF THE
*               STRING AND SYMBOL TABLE THEN AN
*               OVERLAP ERROR HAS OCCURRED AND THE
*               ERROR WORD, ERROR, IS SET TO AN
*               EVEN VALUE.  CONTROL IS THEN PASSED
*               TO ROLRX WHICH IN TURN CALLS PHASE
*               4.  COMPILATION IS DISCONTINUED
*               SINCE EACH SUCCEEDING PHASE DETECTS
*               THE OVERLAP AND PASSES CONTROL TO
*               THE SUCCEEDING PHASE WITHOUT
*               EXECUTING UNTIL PHASE 21 IS LOADED.
*               PHASE 21 EXECUTES AND PRINTS AN
*               OVERLAP ERROR MESSAGE.
*               INCASE OF NORMAL COMPILATION ERRORS
*               THE NORMAL COMPILATION FLOW IS NOT
*               INTERRUPTED.  AN ERROR MESSAGE
*               REPLACES THE ERRONEOUS STATEMENT
*               ON THE STRING AND COMPILATION
*               CONTINUES.
*TABLES/WORK AREAS-
*    NON EXCEPT STRING STRING, SYMBOL TABLE AND
*    FORTRAN COMMUNICATION AREA
*ATTRIBUTES-N/A
*NOTES
*    * THE COMPILATION ERRORS DETECTED IN THIS
*      PHASE ARE NUMBERS 5,6 AND 9.
*    * THE SWITCHES USED IN PHASE 3 FOLLOW.  IF
*      NONZERO, THE SWITCH IS T (TRANSFER). IF THE
*      SWITCH IS ZERO THEN THE SWITCH IS N (NORMAL)
*       * S 1-SUBROUTINE OR FUNCTION ALLOWABLE
*               T=NOT ALLOWABLE
*       * S 2-TYPE STATEMENT ALLOWABLE
*               T=NOT ALLOWABLE
*       * S 3-DIMENSION STATEMENT ALLOWABLE
*               T=NOT ALLOWABLE
*       * S 4-COMMON STATEMENT ALLOWABLE
*               T=NOT ALLOWABLE
*       * S 5-EQUIVALENCE STATEMENT ALLOWABLE
*               T=NOT ALLOWABLE
*       * S 7-STATEMENT ORDER INDICATOR
*               T=STATEMENT OUT OF ORDER
          ABS       REF   CORE
*         SYSTEM AND FORTRAN EQUATES
MEMRY     EQU       ~~REF~~ CORE MAXIMUM CORE SIZE
PHSIZ     EQU       4*320        MAXIMUM PHASE SIZE
OVERL     EQU       MEMRY-PHSIZ     PHASES 2-29 START
FCOM      EQU       OVERL-22     FORTRAN COMM. TABLE
PHNTB     EQU       FCOM-56      PHASE TABLE
ROLRX     EQU       PHNTB-5      INTERPHASE CALL
*         FORTRAN COMMUNICATION AREA
          ORG       FCOM
SOFS      BSS       2            START OF STRING
EOFS      BSS       1            END OF STRING
SOFST     BSS       -            START OF SYMBOL TABLE
SOFNS     BSS       1            START OF NON-STMNT NUMBERS
SOFXT     BSS       1            START OF SUBSCRIPTED TEMPS
```

```
120  SOFGT BSS     1          START OF GENERATED TEMPS
121  EOFST BSS     1          END OF SYMBOL TABLE
122  COMON BSS     1          NEXT AVAILABLE COMMON
123  CSIZE BSS     1          SIZE OF COMMON
124  ERROR BSS     1          OVERLAP ERROR
125  FNAME BSS     1          PROGRAM NAME
126        BSS     1          2ND WD OF NAME
127  SOPF  BSS     1          SUBR (-) OR FUNC (+)
128  CCWD  BSS     1          CONTROL CARD WORD
129  *                              BIT 15 TRANSFER TRACE
130  *                              BIT 14 ARITHMETIC TRACE
131  *                              BIT 13 EXTENDED PRECISION
132  *                              BIT 12 LIST SYMBOL TABLE
133  *                              BIT 11 LIST SUBPROGRAM NAMES
134  *                              BIT 10 LIST SOURCE PROGRAM
135  *                              BIT  9 ONE WORD INTEGERS
136  IOCS  BSS     1          IOCS IOCS CONTROL CARD WOR
137  *                              BIT 15 CARD
138  *                              BIT 14 PAPER TAPE
139  *                              BIT 13 TYPEWRITER
140  *                              BIT 12 1443 PRINTER
141  *                              BIT 11 MAGNETIC TAPE
142  *                              BIT 10 KEYBOARD
143  *                              BIT  9 1442 PUNCH
144  *                              BIT  8 DISK
145  *                              BIT  7 NOT USED
146  *                              BIT  6 NOT USED
147  *                              BIT  5 NOT USED
148  *                              BIT  4 NOT USED
149  *                              BIT  3 NOT USED
150  *                              BIT  2 PLOTTER
151  *                              BIT  1 UNFORMATTED DISK
152  *                              BIT  0 UNFORMATTED TAPE
153  DFCNT BSS     1          DEFINE FILE COUNT
154  LCOMN BSS     2          INSKEL COMMON SIZE
155  ICWER BSS     2          IOCS CONTROL CARD ERROR
156        BSS     3          SYSTEM LOADER USE
157  *                    END OF FORTRAN COMMUNICATION
158  *                    AREA
159        ORG     OVERL      PHASE EXECUTION ADDR
160  START LD      ERROR      CHECK FOR OVERLAP ERROR
161        BSC  L   EOP,Z      BRANCH AROUND IF ERROR
162  *                    INITIALIZE PHASE
163        LDX   11 SOFS       INITIALIZE I/P STRING PT
164  FIOCL LD    1  /          LOAD STMNT ID WORD
165        SRA      -1         GET STMNT ID TYPE
166        S        FIOC       IS IT INTERNAL OUTPUT FMT
167        BSC  L   MOVE,+-    BRANCH IF YES
168  *                    CHECKS FOR A FUNCTION STMNT
169  STL   LD    1  0          LOAD STMNT ID WORD
170        SRA      1
171        AND      =0TYPE     GET STMNT ID TYPE
172        S        =0FUNCK    IS IT FUNCTION
173        BSC  L   SUBRT,Z    BRANCH IF NOT
174        LD       =1E        SET FUNCTION SWITCH (+)
175        STO      SOPF       IN SOPF
176  JIT   LD       SW1        IS SUBR OR FUNC ALLOWED
177        BSC  L   BO,Z       BRANCH IF NOT
178  *                    DOES STMNT HAVE STMNT NUMBER
179        BDX  L   SW1,1      SET SUBR/FUNC NOT ALLOWED
```

```
180   *                       CHECKS THE STMNT FOR A
181   *                       STMNT NUMBER
182   ABEL  LD     1 0        LOAD STMNT ID WORD
183         BSC  L  MOVE,E     BRANCH IF NUMBERED STMNT
184   *                       TEST SW7
185   JACK  LD        SW7     IS STMNT OUT OF ORDER
186         BSC  L  MOVE,+-    BRANCH IF NOT
187   *                       SET UP ERROR NO. 5
188   ERR   LD        ERR5    LOAD ERROR NO. 5
189         STO       ERRNO   STORE
190   *                       REPLACES THE ERRONEOUS STMNT
191   *                       WITH AN ERROR MESSAGE
192   *                       CLOSES UP THE STRING
193   CLOSE LD     1 0        LOAD STMNT ID WORD
194         SRA      2        SHIFT NORM TO RIGHT
195         AND      CONRM    CLEAR ALL BUT NORM
196         STO      NRMSV
197         STX  1   SAVE1    SAVE INPUT POINTER
198         LD       ERRID    LOAD ERROR STMNT ID WORD
199         STO  1   0        PUT ON STRING
200         LD       EOFS     LOAD END OF STRING ADDRES
201         S        SAVE1    GET RANGE OF MOVE LOOP BY
202         S        NRMSV    SUBTRACTING PRESENT ADDR
203         STO      RANGE    THEN SUBTRACTING PRESENT
204         LDX  13  RANGE    STMNT SIZE
205         MDX  3   1
206         MDX  1   -1       MOVE INPUT POINTER
207         LD       ERRNO    PUT ERROR NO.
208         STO  1   0        ON STRING
209         LD       SAVE1    GET ADDRESS OF
210         A        NRMSV    NEXT STATEMENT
211         STO      GET
212         LDX  12  GET      INITIALIZE POINTER
213   LOOZ  LD     2 0        MOVE WORDS IN STRING
214         STO  1   -1       NEXT TO ERROR STMNT
215         MDX  1   -1       MOVE POINTERS
216         MDX  2   -1
217         MDX  3   -1       DECREMENT RANGE OF LOOP
218         MDX      LOOZ     CONTINUE LOOP
219         STX  1   EOFS     NEW END OF STRING ADDRESS
220         LDX  11  SAVE1    RESET INPUT POINTER
221   *                       SW7 TO NORMAL
222         SLA      16
223         STO      SW7      CLEAR OUT OF ORDER SWITCH
224         BSC  L   MOVE     GO TO NEXT STMNT
225   *                        CONSTANTS
226   FIVE  DC       /1B      INTERNAL OUTPUT FORMAT
227   ONE   DC       1        ONE
228   GET   DC       0        NEXT STMNT ADDR
229   IDTPE DC       /7C00    STMNT ID TYPE MASK
230   FUNCK DC       /3400    FUNCTION
231   SW1   DC       1        SWITCH 1
232   SW7   DC       1        SWITCH 7
233   ERR5  DC       5        ERROR NO. 5
234   ERRNO DC       0        ERROR NO.
235   IDNRM DC       /01FF    NORM MASK
236   NRMSV DC       0        NORM STORAGE
237   SAVE1 DC       0        POINTER
238   ERRID DC       /A008    ERROR STMNT ID WORD
239   RANGE DC       0        RANGE OF LOOP
```

```
240  DIP    DC      0           INPUT POINTER
241  DIP1   DC      0           POINTER
242  SUBRC  DC      /0C00       SUBROUTINE
243  REALC  DC      /2400       REAL
244  SW2    DC      0           SWITCH 2
245  SW3    DC      0           SWITCH 3
246  INTC   DC      /0400       INTEGER
247  DIMC   DC      /2000       DIMENSION
248  COMMC  DC      /1C00       COMMON
249  TWO    DC      2           TWO
250  MINUS  DC      /FFFF       MINUS
251  VALYU  DC      9           ELIMINATES STMNT NO. BITS
252  DFILE  DC      /7800-/[illegible]   DEFINE FILE
253  DATAS  DC      /7C00-/[illegible]   DATA
254  OKSW   DC      *-*         COMMON OUT OF ORDER
255  *                    REMOVES THE STMNT NO. FROM
256  *                    NUMBERED SPECIFICATION STMNTS,
257  *                    ADJUSTS THE NORM
258  REMOVE STX     1 DIP         SAVE INPUT POINTER
259         MDX     1 3           MOVE POINTER TO STMNT BOD
260         STX     1 DIP1        SAVE INPUT POINTER
261         LDX     2 DIP1
262         LD      L EOFS        LOAD END OF STRING
263         S         DIP         GET RANGE OF MOVE LOOP
264         S         TWO
265         STO       RANGE
266         LDX     3 RANGE       INITIALIZE RANGE COUNTER
267         MDX     1 -3          RESTORE INPUT POINTER
268  LOOP   LD      2 0           MOVE WORD DOWN
269         STO     1 0           TO NEW POSITION
270         MDX     1 1           MOVE POINTERS
271         MDX     2 1
272         MDX     3 -1          DECREMENT RANGE OF MOVE
273         MDX       LOOP        CONTINUE LOOP
274         STX   L1 EOFS         NEW END OF STRING ADDRESS
275         LDX   L1 DIP          RESTORE INPUT POINTER
276         LD      1 0           REMOVE STMNT NO. BIT AND
277         S         VALYU       ADJUST NORM IN ID WORD
278         STO     1 0           PUT BACK ON STRING
279         MDX       JACK        RETURN
280  *                    CHECKS FOR A SUBROUTINE STMNT
281  SUBRT  S         SUBRC       IS IT SUBROUTINE
282         BSC   L  TENT,Z       BRANCH IF NOT
283         LD       MINUS        SET SUBROUTINE SWITCH (-)
284         STO   L  SUBF         IN SUBF
285         MDX      JIM          GO CHECK SWITCH 1
286  TBIT   MDX   L  SW1,1        SET SUBR/FUNC NOT ALLOWED
287  *                    CHECKS FOR DEFINE FILE STMNTS
288         S         DFILE       IS IT DEFINE FILE
289         BSC   L  TSTOK,+-     BRANCH IF YES
290         S         DATAS       IS IT DATA
291         BSC   L  CKRL,Z       BRANCH IF NOT
292  TSTOK  LD       SW5          IS EQUIVALENCE STMNT
293         BSC   L  GO,Z         ALLOWED - BRANCH IF NOT
294         LD       SW4          IS COMMON STMNT ALLOWED
295         BSC   L  ABEL,+-      BRANCH IF YES
296         STX     0 OKSW        SET COMMON OUT OF ORDER S
297         BSC   L  ABEL         CHECK FOR STMNT NO
298  *                    CHECKS FOR REAL STMNTS
299  CKRL   LD      1 0           LOAD STMNT ID WORD
```

```
               SRA         1
               AND         IDTPE       GET STMNT ID TYPE
               STO         MAM         STORE ID TYPE
               S           REALC       IS IT REAL
               BSC   L     TENT1,Z     BRANCH IF NOT
      SON      LD          SW2         IS TYPE STMNT ALLOWABLE
               BSC   L     LABEL,+-    BRANCH IF YES
      BO       MDX   L     SW7,1       SET STMNT OUT OF ORDER SW
               BSC   L     LABEL       CHECK FOR STMNT NO.
      *                           CHECKS FOR INTEGER, EXTERNAL,
      *                           AND DIMENSION STMNTS
      TENT1    S           INTC        IS IT INTEGER
               BSC   L     SON,+-      BRANCH IF YES
               LD          MAM         LOAD STMNT ID TYPE
               S           EXTER       IS IT EXTERNAL
               BSC   L     SON,+-      BRANCH IF YES
               MDX   L     SW2,1       SET TYPE NOT ALLOWABLE SW
               LD        1 0           LOAD STMNT ID WORD
               SRA         1
               AND         IDTPE       GET STMNT ID TYPE
               S           DIMC        IS IT DIMENSION
               BSC   L     TAG3,Z      BRANCH IF NOT
               LD          SW3         IS DIM STMNT ALLOWABLE
               BSC   L     LABEL,+-    BRANCH IF YES
               MDX         BO          SET STMNT OUT OF ORDER SW
      TAG3     MDX   L     SW3,1       SET DIM NOT ALLOWABLE SW
      *                           CHECKS FOR COMMON STMNTS
               LD        1 0           LOAD STMNT ID WORD
               SRA         1
               AND         IDTPE       GET STMNT ID TYPE
               S           COMMC       IS IT COMMON
               BSC   L     TAG4,Z      BRANCH IF NOT
               LD          SW4         IS COMMON STMNT ALLOWED
               BSC   L     LABEL,+-    BRANCH IF YES
               MDX         BO          SET STMNT OUT OF ORDER SW
      *                           CONSTANTS
      MAM      DC          0           ID STORAGE
      EXTER    DC          /6400       EXTERNAL
      SW4      DC          0           SWITCH 4
      EQUIC    DC          /5400       EQUIVALENCE
      SW5      DC          0           SWITCH 5
      CONTC    DC          /5800       CONTINUE
      DIM2     DC          0           TEMPORARY STORAGE
      DIM3     DC          0           TEMPORARY STORAGE
      GOTOC    DC          /E000       GO TO
      TAG4     MDX   L     SW4,1       SET COMMON NOT ALLOWED SW
      *                           CHECKS FOR EQUIVALENCE STMNTS
               LD        1 0           LOAD STMNT ID WORD
               SRA         1
               AND   L     IDTPE       GET STMNT ID TYPE
               S           EQUIC       IS IT EQUIVALENCE
               BSC   L     TAG5,Z      BRANCH IF NOT
               LD          SW5
               OR          OKSW        IS EQUIVALENCE STMNT
               BSC   L     LABEL,+-    ALLOWED - BRANCH IF YES
               MDX         BO          SET STMNT OUT OF ORDER SW
      TAG5     MDX   L     SW5,1       SET EQUIVALENCE NOT ALLOW
      *                           CHECKS FOR CONTINUE STMNTS
               LD        1 0           LOAD STMNT ID WORD
               SRA         1
```

```
          AND       IDSTY     GET STMNT ID TYPE
          S         CONTC     IS IT CONTINUE
          BSC  L    EFF,Z     BRANCH IF NOT
          LD    1   0         LOAD SWITCH ID WORD
          BSC  L    MOVE,E    BRANCH IF HAVE STMNT NO.
          BSI       RMOV1     REMOVE WORD FROM STRING
          BSC  L    ST1       CHECK STATEMENT
*                         UPDATES THE STRING I/P PT (XR1)
*                         TO MOVE TO THE NEXT STMNT
MOVE      LD    1   0         LOAD STMNT ID WORD
          SRA       2         SHIFT RIGHT TO GET NORM
          AND  L    IDNRM     GET NORM
          STO       NXID+1
NXID      MDX  L1             GET ADDR OF NEXT STMNT
          BSC  L    FIOCL     SEE IF FIO STMNT
*                         REMOVE STATEMENT FROM STRING
RMOV1     DC        *-*       RETURN ADDRESS
          STX   1   DIP2      SAVE POINTER
          SRA       2         SHIFT NORM TO RIGHT
          AND  L    IDNRM     CLEAR ALL BUT NORM
          STO  L    NRMSV     ADD NORM TO POINTER
          A         DIP2      TO GET ADDRESS OF
          STO       DIP3      NEXT STATEMENT
          LDX  I2   DIP3      INITIALIZE LOOP INPUT PT
          LD   L    EOFS      GET RANGE OF LOOP BY
          S         DIP2      SUBTRACTING POINTER FROM
          S    L    NRMSV     END OF STRING THEN
          STO       DIP3      SUBTRACT NORM
          LDX  I3   DIP3      INITIALIZE RANGE COUNTER
          MDX   1   -1
          MDX   3   1
LOOP1     LD    2   0         MOVE WORD DOWN
          STO   1   1         TO NEW POSITION
          MDX   1   1         MOVE POINTERS
          MDX   2   1
          MDX   3   -1        DECREMENT RANGE COUNTER
          MDX       LOOP1     CONTINUE LOOP
          STX  L1   EOFS      NEW END OF STRING ADDRESS
          LDX  I1   DIP2      RESTORE INPUT POINTER
          BSC  I    RMOV1     RETURN
ER36      DC        6         ERROR NO. 6
IFC       DC        /0400     IF
ENDC      DC        /0800-/4000  END
RTNCN     DC        /4000-/5000  RETURN
STOPC     DC        /2000     STOP
ERCN      DC        /14-/02   ERROR
CLKST     DC        /7000-/7000  CALL LINK
CEXST     DC        /7001-/7000  CALL EIXT
CLLST     DC        /1800-/7001  CALL
CONT      DC        /8004     CONTINUE
IDSTY     DC        /7C01     ID MASK
*                         CHECKS FOR THE PRESENCE OF
*                         TRANSFER STATEMENTS
EFF       S         GOTOC     IS IT GO TO
          BSC  L    MOVES,+-  BRANCH IF YES
          S         IFC       IS IT IF
          BSC  L    MOVES,+-  BRANCH IF YES
          S         STOPC     IS IT STOP
          BSC  L    MOVES,+-  BRANCH IF YES
          S         RTNCN     IS IT RETURN
```

```
          BSC  L  MOVES,+-   BRANCH IF YES
          S       ENDC       IS IT END
          BSC  L  INIT,+-    BRANCH IF YES
          S       CLKST      IS IT CALL LINK
          BSC  L  MOVES,+-   BRANCH IF YES
          S       CETST      IS IT CALL EXIT
          BSC  L  MOVES,+-   BRANCH IF YES
          MDX     MOVE       GO TO NEXT STMNT
*                        MOVE TO NEXT STATEMENT
MOVES LD      1 0            LOAD STMNT ID WORD
      SRA       2            SHIFT NORM TO RIGHT
      AND  L  DNRM           GET NORM
      STO     NEXT+1
*                       CHECKS FOR STMNT NUMBERS IN
*                       STMNTS FOLLOWING TRANSFER
*                       STMNTS
NEXT  MDX  L1                GET ADDRESS OF NEXT STMNT
CKEND LD      1 0            LOAD STMNT ID WORD
      SRA       11           GET STMNT ID TYPE
      S    L  TWO            IS IT END
      BSC  L  INIT,+-        BRANCH IF YES
      S       ERCN           IS IT ERROR
      BSC  L  ST1,+-         BRANCH IF YES
*                       DOES STATEMENT HAVE NUMBER
      LD      1 0            LOAD STMNT ID WORD
      BSC  L  ST1,E          BRANCH IF HAS STMNT NO.
      S       CONT           IS IT CONTINUE
      BSC  L  STER6,Z        BRANCH IF NOT
      LD      1 0            LOAD STMNT ID WORD
      BSI  L  RMOV1          REMOVE STMNT FROM STRING
      BSC  L  CKEND          CHECK FOR END
*                        SET UP ERROR NO. 6
STER6 LD      ERR6           LOAD ERROR NO. 6
      STO  L  ERRNO          STORE IN ERROR NO.
      BSC  L  CLOSE          CLOSE UP STRING
*                 INITIALIZE PHASE
*                       INITIALIZES THE PHASE
*                       CHECKS FOR A PREVIOUS OVERLAP
*                       ERROR
INIT  LDX  I1 SOFS           INITIALIZE INPUT POINTER
      LD   L  ERROR          CHECK FOR OVERLAP ERROR
      BSC  L  EOP,Z          BRANCH ON OVERLAP ERROR
*                       CHECKS FOR THE END STMNT
*                       CHECKS FOR A STMNT NO. IN STMNT
*                       OTHER THAN END
ENDST LD      1              LOAD STMNT ID WORD
      AND     IDCL           GET STMNT ID TYPE
      S       IDEND          IS IT END
      BSC  L  EOP,+-         BRANCH IF YES
      LD      1              DOES STMNT HAVE STMNT NO.
      BSC  L  LOOK,E         BRANCH IF YES
*                 MOVE TO NEXT STATEMENT
MOVS  LD      1 0            LOAD STMNT ID WORD
      AND     NMCL           GET NORM
      SRA     2              MOVE NORM TO RIGHT
      STO     A+1
A     MDX  L1 0              GET ADDR OF NEXT STMNT
      MDX     ENDST          CHECK FOR END STMNT
*                 BRANCHES TO ROLRX ROUTINE TO LOAD
*                       THE NEXT PHASE
```

```
480  EOF   BSI  L  POLRX     CALL TO GET PHASE 4
481        DC       04       NEXT PHASE NUMBER
482  *                  REPLACES THE ERRONEOUS STMNT
483  *                  WITH AN ERROR MESSAGE
484  *                  CLOSES UP THE STRING
485  CLOZE STO    1 2        STORE ERROR NUMBER
486        LD     1          LOAD STMNT ID WORD
487        AND       NMCL    CLEAR ALL BUT NORM
488        SRA       2       SHIFT NORM TO RIGHT
489        STO       *+1
490        LD        ERTY    LOAD ERROR STMNT ID WORD
491        STO    1          PUT ON STRING
492        STX    1 TEMP     STORE INPUT POINTER
493        LDX    12 TEMP    INITIALIZE LOOP OUTPUT PT
494        MDX    2 2        MODIFY PT FOR STMNT SIZE
495        LDX    13 TEMP    INITIALIZE LOOP INPUT PT
496  C     MDX    L3         MODIFY ID TO GET ADDRESS OF
497        MDX    3 -1       NEXT STATEMENT
498        STX    3 TEMP
499        STX    1 TMKOK    SAVE INPUT POINTER
500        LD     L EOFS     GET RANGE OF LOOP BY
501        S         TEMP    SUBTRACTING POINTER FROM
502        STO       TEMP    END OF STRING
503        LDX    L1 TEMP    LOAD RANGE OF LOOP
504  LOOPB MDX    3 -        RESET INPUT POINTER
505        LD     3          MOVE WORD DOWN
506        STO    2          IN STRING
507        MDX    2 2        MOVE POINTER
508        MDX    1 -1       DECREMENT RANGE OF LOOP
509        MDX       LOOPB   CONTINUE LOOP
510        LDX    L1 TMKOK   RESET INPUT POINTER
511        MDX    2 -1
512        STX    L2 EOFS    NEW END OF STRING ADDRESS
513        MDX       MOVS    GO TO NEXT STATEMENT
514  *                  CONSTANTS AND WORKING STORAGE
515  IDCL  DC         /F800   ID MASK
516  IDEND DC         /1000   END
517  NMCL  DC         /07FC   NORM MASK
518  TEMP  DC         0       TEMPORARY STORAGE
519  STMOC DC         /0200   STMNT NO. SYMBOL TABLE ID
520  THREE DC         3       THREE
521  SIGN  DC         /8000   USEFUL CONSTANT
522  TMKOK DC         0       TEMPORARY STORAGE
523  ERTY  DC         /A008   ERROR STMNT ID WORD
524  C2    DC         2       TWO
525  C9    DC         9       NINE
526  *                  SCANS THE SYMBOL TABLE FOR A
527  *                  DUPLICATE STMNT NUMBER
528  LOOK  LDX    12 SOFST     INITIALIZE SYMBOL TABLE P
529        MDX       RETRY+1   SKIP NEXT INSTRUCTION
530  RETRY MDX    2 -3         GO TO NEXT TABLE ENTRY
531        STX    2 TEMP       SAVE POINTER
532        LD        TEMP      HAS ENTIRE SYMBOL TABLE
533        S      L  EOFST     BEEN CHECKED FOR DUPLICAT
534        BSC    L  PUTIN,+   BRANCH IF YES
535        LD     2 1          COMPARE FIRST WORD
536        S      1 1          OF NAMES - IF UNEQUAL
537        BSC    L  RETRY,Z   GET NEW TABLE ENTRY
538        LD     2 2          COMPARE SECOND WORD OF
539        S      1 2          NAMES - IF UNEQUAL GET NE
```

```
          BSC  L  RETRY,Z   SYMBOL TABLE ENTRY
*                       SET UP ERROR NO. 9
          LD        C9      LOAD ERROR NO.
          MDX       CLOZE   CLOSE UP STRING
*                      PLACES THE STMNT NO. INTO THE
*                      SYM TBL - UPDATES THE FORTRAN
*                      COMMUNICATIONS AREA TO REFLECT
*                      CHANGES IN THE TABLE*S LENGTH -
*                      REPLACES THE STRING AREA STMNT
*                      NUMBER WITH THE SYMBOL TABLE
*                      ADDR WHERE IT IS NOW LOCATED
PUTIN     LD     1  1       LOAD 1ST WORD IN NAME
          STO    2  1       PUT IN SYMBOL TABLE
          LD     1  2       LOAD 2ND WORD IN NAME
          STO    2  2       PUT IN SYMBOL TABLE
          LD        STNOC   LOAD SYMBOL TABLE ID WORD
          STO    2          PUT IN SYMBOL TABLE
          MDX    2  -3      MOVE SYMBOL TABLE POINTER
          STX   L2  SOFNS   NEW START OF NON-STMNT-NO
          STX   L2  EOFST   NEW END OF SYMBOL TABLE
          MDX    L  SOFXT,-3  NEW START OF SUBSCR-TEMP
          MDX    L  SOFGT,-3  NEW START OF GENER-TEMP
*                   PUT RELATIVE ADDRESS IN STRING
          LD     L  SOFST   GET SIZE OF SYMBOL TABLE
          S      L  EOFST
          SRT       16      SHIFT FOR DIVIDE
          D         THREE   GET NO. OF ENTRIES IN TBL
          OR        SIGN    PUT IN SIGN
          STO    1  1       STORE IN STRING
          STX    1  B+1     SAVE INPUT POINTER
B         MDX    L  *,-4    MODIFY NORM BY -1
          LDX   I3  B+1     INITIALIZE POINTER
          MDX    3  2       MOVE POINTER
          LD     L  EOFS    GET RANGE OF MOVE LOOP
          S         B+1
          STO       TEMP
          LDX   I2  TEMP    INITIALIZE RANGE COUNTER
          MDX    2  -2      MODIFY RANGE
LOOPA     LD     3  1       MOVE WORD DOWN
          STO    3          IN STRING
          MDX    3  1       MOVE POINTER
          MDX    2  -1      DECREMENT RANGE
          MDX       LOOPA   CONTINUE LOOP
          MDX    L  EOFS,-1  NEW END OF STRING ADDRESS
*                   SYMBOL TABLE OVERLAP
          LD     L  EOFS    LOAD END OF STRING
          S      L  EOFST   CHECK FOR OVERLAP BY SUBT
          S      L  C2      LAST SYMBOL TABLE ENTRY
          BSC    L  NOVS,+  BRANCH NO OVERLAP
*                   SET UP OVERLAP ERROR
          MDX    L  ERROR,1  SET ERROR INDICATOR
          BSC    L  EOP     GO TO END OF PHASE
          BSS       OVERL-*-32.*2  PHASE-3 PATCH AREA
          END       START
```