```
1  *    GENERAL AUTOMATION, INC.  ALL RIGHTS RESERVED
2  ************************************************
3  *
4  *    PROGRAM NAME  FPH-26
5  *
6  *    MODEL NUMBER  8F026
7  *
8  *    PURPOSE       FORTRAN PHASE-26
9  *
10 *    PROGRAMMER    DICK WALLMANN
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 ** OUTPUT I
22 ************************************************
23 *STATUS-VERSION 1, MODIFICATION 0
24 *
25 *FUNCTION/OPERATION-
26 *    * BUILD  PROGRAM HEADER AND DATA HEADER
27 *      RECORDS
28 *    * ANALYZES DEFINE FILE STATEMENTS AND PUTS
29 *      THEM IN THE BUFFER FOR OUTPUT TO WORKING
30 *      STORAGE IN ABSOLUTE MODE  EXCEPT FOR THE
31 *      ASSOCIATED VARIABLE WHICH IS IN RELOCATABLE
32 *      MODE
33 *    * SCANS THE SYMBOL TABLE FOR REAL AND INTEGER
34 *      CONSTANTS AND PLACES THEM IN THE BUFFER IN
35 *      ABSOLUTE MODE FOR OUTPUT TO WORKING STORAGE
36 *
37 *ENTRY POINTS-
38 *    * NEQ - PHASE 26 IS LOADED BY PHASE 25 VIA
39 *            THE ROLRX ROUTINE, CONTROL IS PASSED,
40 *
41 *INPUT-THE STATEMENT STRING AND SYMBOL TABLE FROM
42 *    THE PREVIOUS PHASES
43 *
44 *OUTPUT-
45 *    * 54 WORD CARD IMAGE RECORDS
46 *    * A 54  WORD BUFFER CONTAINING SIMILAR
47 *      INFORMATION TO BE OUTPUT TO WORKING STORAGE
48 *      OR TO BE PASSED ON TO PHASE 27 TO BE FILLED
49 *      AND PUT INTO WORKING STORAGE
50 *
51 *EXTERNAL REFERENCES-
52 *    * SUBROUTINES-
53 *      ROLRX - PHASE ROLLING ROUTINE
54 *      LIO - IOCS ENTRY
55 *    * OTHER FORTRAN PHASES
56 *      NONE
57 *
58 *EXITS-
59 *    * NORMAL
```

```
60  *          PHASE 27 IS LOADED VIA ROLRX AND CONTROL
61  *          IS PASSED TO IT.
62  *    * ERRORS-
63  *      OVERLAP-
64  *          IF A PREVIOUS PHASE HAS DETECTED AN
65  *          OVERLAP ERROR THEN AN IMMEDIATE EXIT
66  *          OTHERWISE, NO OVERLAP ERROR IS DETECTED.
67  *      SYNTAX-
68  *          IF SYNTAX ERRORS HAVE OCCURRED THEN
69  *           OUTPUT IS SUPPRESSED,  PHASE
70  *          27 IS CALLED VIA ROLRX.
71  *
72  *TABLES/WORK AREAS-
73  *    * THE STATEMENT STRING
74  *    * THE SYMBOL TABLE
75  *    * THE FORTRAN COMMUNICATIONS AREA
76  *
77  *NOTES-
78  *    PROGRAM HEADER CONSISTS OF 12 WORDS IN THE
79  *    BUFFER AREA
80  *       WD 1 RESERVED
81  *       WD 2 RESERVED
82  *       WD 3 INTEGER TYPE, PRECISION
83  *       WD 4 OBJECT TIME PROGRAM LENGTH
84  *       WD 5 LENGTH OF COMMON
85  *       WD 6 WD SIZE OF PROGRAM HEADER = 9   3
86  *       WD 7 VAR. AREA WHEN NO DEFINE FILE STMNT
87  *       WD 8 LENGTH OF PROGRAM IN DISK BLOCKS
88  *       WD 9 NUMBER OF FILES DEFINED
89  *       WD10 PROGRAM NAME
90  *       WD11 PROGRAM NAME
91  *       WD12 EXECUTION ADDRESS
92  *
93          ABS   REF CORE
94  *
95  *      SYSTEM AND FORTRAN EQUATES
96  *
97  MEMRY EQU     ~~17FFF~~ CORE    MAXIMUM CORE SIZE
98  PHSIZ EQU     4*320             MAXIMUM PHASE SIZE
99  OVERL EQU     MEMRY-PHSIZ       PHASES 2-29 START
100 FCOM  EQU     OVERL-22          FORTRAN COMM. TABLE
101 PHNTB EQU     FCOM-56           PHASE TABLE
102 ROLRX EQU     PHNTB-50          INTERPHASE CALL
103 *
104 LIO   EQU     /85               IOCS ENTRY
105 *
106 *     FORTRAN COMMUNICATIONS AREA
107 *
108         ORG     FCOM
109 SOFS  BSS     1          START OF STRING
110 EOFS  BSS     1          END OF STRING
111 SOFST BSS     1          START OF SYMBOL TABLE
112 SOFNS BSS     1          PROG LENGTH AT OBJECT TIME
113 SOFXT BSS     1          SIZE OF WORK AREA
114 SOFGT BSS     1          SIZE OF CONSTANTS AREA
115 EOFST BSS     1          END OF SYMBOL TABLE
116 COMON BSS     1          ENTRY POINT
117 CSIZE BSS     1          SIZE OF COMMON
118 ERROR BSS     1          OVERLAP ERROR
119 FNAME BSS     1          PROGRAM NAME
```

```
120         BSS     1           2ND WORD PROG NAME
121   SORF  BSS     1           SUBR (-) OR FUNC (+)
122   CCWD  BSS     1           CONTROL CARD WORD
123   *                 BIT 15 TRANSFER TRACE
124   *                 BIT 14 ARITHMETIC TRACE
125   *                 BIT 13 EXTENDED PRECISION
126   *                 BIT 12 LIST SYMBOL TABLE
127   *                 BIT 11 LIST SUBPROGRAM NAMES
128   *                 BIT 10 LIST SOURCE PROGRAM
129   *                 BIT  9 ONE WORD INTEGERS
130   IOCS  BSS     1           IOCS CONTROL CARD WORD
131   *
132   *        SEE PHASE ONE FOR BIT PATTERNS
133   *
134   DFCNT BSS     1           DEFINE FILE COUNT
135   *
136   LCOMN BSS     2           INSKEL COMMON
137   *
138   ICCER BSS     2           IOCS CONTROL CARD ERROR
139   *
140         BSS     2           SYSTEM LOADER USE
141   *
142   *                   END OF FORTRAN COMMUNICATION
143   *                   AREA
144   ****************************************************
145   *                 THE SWITCHES USED IN PHASE 26 FOLLOW
146   *                 IF POSITIVE, THE SWITCH IS TRANSFER
147   *                 IF ZERO, THE SWITCH IS NORMAL N
148   *                 SWITCH COSW
149   *                   N  OUTPUT OF REAL CONSTANTS
150   *                   T  OUTPUT OF INTEGER CONSTANTS
151   *                 SWITCH TWOWC
152   *                   N  OUTPUT OF TWO-WORD CALL NAME
153   *                   T  OUTPUT OF ONE-WORD CALL NAME
154   *
155   *
156   *
157   *                 BUFFER COMMUNICATIONS AREA
158   *                 AND BUFFER
159   *
160   *                 RESERVED FOR COMMUNICATION
161   *                 BETWEEN PHASES
162   *                 OUTPUT1 AND OUTPUT2
163   *
164         ORG       OVERL+3*320     TO BUFFER AREA
165         BSS  E    0
166   BUFPT DC        ***             BUFFER POINTER
167   INPT  DC        ***             INDICATOR POINTER
168   INCT  DC        ***             INDICATOR COUNTER
169   LOCTR DC        0           LOCATION COUNTER
170   RECCT DC        1            RECORD COUNT
171   *               OUTPUT BUFFER
172   BUFF  DC        54              BUFFER WORD COUNT
173   WD1   DC        0
174   WD2   DC        0           RESERVED
175   WD3   DC        0           TYPE, PRECISION
176   WD4   DC        0                 INSKELL COMMON SIZE
177   WD5   DC        0           LENGTH OF COMMON
178   WD6   DC        3           WD SIZE OF PROG HDR = 9
179   WD7   DC        0           VAR AREA IF NO DEFINE FILES
```

```
180          DC      0
181  WD9     DC      0          NUMBER OF FILES DEFINED
182  WD10    DC      0          PROGRAM NAME
183          DC      0          WORD 2 OF PROGRAM NAME
184  WD12    DC      0          EXECUTION ADDR
185  *
186  *
187  *            FOLLOWING INITIAL PROGRAM
188  *            IS WITHIN BUFFERAREA
189  *
190  START LD   L  FNAME            PROGRAM NAME
191        STO  L  WD10
192        LD   L  FNAME+1
193        STO  L  WD10+1
194        LD   L  LCOMN        GET COMMON SIZE
195        STO  L  WD4          PUT IT IN HEADER WORD 4
196  *
197        LDX  L3 ZERO       SETUP IDX FOR CONSTANTS
198  *
199  *     INITIALIZE INDICATORS IN HEADER WORD 9
200        LD   L  DFCNT              GET DEFINE FILE COUNT
201        SRT     16           PUT IT IN Q
202        D     3 SIX-Z        DIVIDE BY SIX (6 WD ENTS)
203        STO     WD9          PUT QUOTIENT IN HDR WD9
204  *
205  *     SET MPX INDICATORS IN HEADER WORD 7
206  *
207        LD   L  IOCS               GET IOCS RECORD WORD
208        RTE     31           PUT BIT 1 IN BIT 0
209        SRA     15           MOVE BIT 1 IN POSITION 15
210        SLT     15           MOVE BIT 0 INTO POSITION 1
211        SRA     15           MOVE BITS TOGETHER
212        SRT     2            BITS NOW REVERSED AND IN Q
213        A     3 ONE-Z         SET ACC BIT 15 ON
214        SRT     1            PUT IN Q TO IND MPX PROG
215        LD    3 H4000-Z      SET FLAG
216        LD   L  IOCS         GET I/O CONT WD
217        SRA     12           PUT PLOTTER BIT IN BIT 15
218        SRT     1            SHIFT IT TO THE Q
219        LD   L  IOCS         GET I/O CONT WD
220        SRA     5            PUT KEYBOARD BIT IN 15
221        SRT     1            SHIFT INTO THE Q
222        LD   L  IOCS         GET I/O CONT WD
223        SRA     4            PUT MAG TAPE BIT IN 15
224        SRT     1            SHIFT INTO THE Q
225        LD   L  IOCS         GET I/O CONT WD
226        SRA     1            PUT PAPER TAPE BIT IN 15
227        SRT     1            SHIFT IT INTO THE Q
228        LD   L  IOCS         GET I/O CONT WD
229        SRA     3            PUT 1443 PRINTER BIT IN 1
230        SRT     1            SHIFT IT INTO THE Q
231        LD   L  IOCS         GET I/O CARD BIT 15
232        SRT     1            SHIFT BIT 15 INTO Q
233        LD   L  IOCS         GET I/O CONT WD
234        SRA     2            PUT TYPEWRITER BIT IN 15
235        SRT     1            SHIFT IT INTO THE Q
236        LD   L  CCWD         GET CONT CARD WD
237        SLA     13           PUT EXTENDED PRECISION
238        SRA     15           BIT IN BUT POSITION 15
239        A     3 ONE-Z        SET STD OR EXTENDED PRECI
```

```
240          SRT      2            ION,  SHIFT INTO THE Q
241          LD    L  CCWD         GET CONT CARD WORD
242          SRA      6            PUT ONE WORD INTEGERS
243          SLT      15           BIT INTO BIT POSITION 15
244          SRT      15           MOVE Q BITS OVER
245          A     3  TWO-Z        TURN BIT 14 OF ACC ON
246          SRT      3            PUT IT IN Q
247          SLT      16           WORD 7 IS NOW FORMED SHIF
248     *                          IT INTO ACC AND STORE V1M
249          STO   L  WD7          STORE IT IN HEADER FO V1M
250     *                          IT INTO ACC AND STORE V1M
251          LD    L  IOCS         GET IOCS RECORD       V1M
252          SLA      8            REMOVE BITS 0 THRU 7
253          SRA      15           REMOVE BIT 9
254          SLA      1            POSITION FOR HDR WD7
255          OR    L  WD7          JOIN THEM
256          STO   L  WD7          STORE IT IN HEADER FO V1M
257          BSC   L  Q1011
258     *
259          ORG      OVERL
260     NEQ  BSC   L  ENT
261     *
262     *               DETERMINE WHETHER MAIN LINE
263     *               OR SUBPROGRAM
264     *
265     Q1011 LD   L  SORF
266          BSC   L  INISR,Z   BR IF SUBROUTINE
267     *
268     *               INITIALIZE MAIN LINE PROGR
269          LD     3 H0200-Z    GET MAIN LINE IND CONST,
270     Q101X STO   L  WD3        SAVE IN 3RD WORD OF PROG HD
271          LD    L  CSIZE      SIZE OF COMMON
272          STO   L  WD5        SAVE IN 5TH WORD OF PROG HD
273     *
274     Q101A LD   L  COMON       PROGRAM ENTRY POINT
275          STO   L  WD12       PUT IN 12TH WORD OF PROG HD
276     *
277          LDX    3 42
278          SLA      17
279          STO   L3 WD12             CLEAR BUFFER REMAINDE
280          MDX    3 -1
281          MDX      *-4              LOOP
282          BSI   L  WRITE            WRITE HEADER RECORD
283     *
284     *               TEST IF ANY  DEFINEFILE  STATEMENTS
285     *               INCLUDED
286     *
287          LD    L  DFCNT      CNT OF NO, DEFINING FILE
288          BSC   L  F1012,Z    BR IF DEFN FILES ARE PRESEN
289          MDX      F2011            BRANCH IF NOT
290     *
291     *               OUTPUT  DEFINE FILE  STATEMENTS
292     COUNT DC       0          NO, WDS DEFN FILE STMNTS
293     F1012 STO      COUNT      SET DEF FILE WORDCOUNTER
294          SLA      16         CLEAR ACC
295          LDX   I1 SOFS       INIT STRING POINTER
296     *
297     F1014 LD       COUNT      NO, WDS DEFN FILE
298     *               SCAN STRING FOR DEFINE FILE STMNTS
299          BSC   L  F2011,     BR IF NO MORE DEFINE FILE
```

```
300  F1015 LD     1 0          GET ID WORD FROM STRING
301        AND    3 H07FC-Z    MASK ALL BUT NORM BITS
302        STO    3 NORM-Z     SAVE NORM IN TEMP STO
303        LD     1 0          GET ID WORD FROM STRING
304        AND    3 HF800-Z    MASK ALL BUT TYPE CODE
305        S      3 HF000-Z    TEST FOR DEFN FILE CODE
306        BSC  L F1021, -     BR IF DEFINE FILE STATEMENT
307  *
308  *                MOVE POINTER TO NEXT STATEMENT
309  *                MOVE POINTER TO NEXT STATEMENT
310        LD     3 NORM-Z     GET STMNT NORM
311        SRA      2          RIGHT JUSTIFY NORM
312        STO      F1016 1    SAVE NORM COUNT
313  F1016 MDX  L1 *-*          INCR I/P STRING PT BY NORM
314        MDX      F1015      BR TO CHECK NEXT STMNT TYPE
315  *
316  *                OUTPUT A DEFINE FILE STATEMENT
317  *
318  *                INITIALIZE COUNTER TO OUTPUT
319  *                FOURTH WORD IN EACH SIXWORD GROUP
320  *                AS RELATIVE SYMBOL TABLE ADDRESS
321  *
322  F1021 LD     3 FOUR-Z     GET CONSTANT   4
323        STO    3 SCT-Z      SET UP COUNTER ON NO. WORDS
324  *
325  F1022 MDX    1 1          INCR POINTER
326        MDX  L  NORM,-4     DECR NORM BY ONE WORD
327        MDX      F1023      BR IF NO SKIP
328        MDX      F1014      BR IF END OF STATEMENT
329  *
330  *                TEST IF FOURTH WD IN GROUP
331  F1023 MDX  L  SCT,-1      DECR WORD COUNTER
332        MDX      F1024      BR IF NO SKIP
333  *
334  *                OUTPUT RELTV ADDR ASSOCIATED VAR-WD
335        LD     1 0          PICK UP FOURTH WORD STMNT
336        BSI    3 GETST-Z    GET SYMBOL TABLE ADDRESS
337        BSI    3 OUREL-Z    OUTPUT RELATIVE WORD
338  *                           OR ABSOLUTE IF IN
339  *                          COMMON
340        LD     3 SIX-Z      RESET CTR ON DEFN FILE WDCN
341        STO    3 SCT-Z      SAVE COUNT IN TEMP CTR
342        MDX      F1025      BR TO CONTINUE O/P OF STMNT
343  *
344  *                OUTPUT ABSOLUTE
345  F1024 LD     1 0
346        BSI    3 OUABS-Z    GO OUTPUT WORD
347  *
348  *                COUNT DEFINE FILE WORDS
349  F1025 MDX  L  COUNT,-1
350        SLA      0          TO BALANCE SKIP
351        MDX      F1022      BR TO CONT DEF FILE TBL O/
352  *
353  *
354  *                HERE COMES OUTPUT OF
355  *                 DATA -STATEMENTS
356  *                SEARCH THROUGH STRING FOR DATA
357  *                STATEMENTS AND OUTPUT
358  *
359  F2011 LDX  I1 SOFS           XR1 = STMT, START
```

```
360   *
361   *                STORE STMNT-ID COMPUTE
362   *                ADDR OF NEXT STATEMENT
363   *
364   DT102 LD     1 0         LOAD STMNT ID WORD
365         AND    3 HF800-Z   MASK OUT ALL BUT STMNT TYPE
366         STO      STID      SAVE STMNT TYPE
367         LD     1 0         LOAD STMNT ID WORD AGAIN
368         AND    3 H07FC-Z   MASK OUT ALL BUT STMNT NORM
369         SRA      2         RIGHT JUSTIFY NORM
370         STX    1 TMP1      SAVE PT OF CURRENT STMNT
371         A        TMP1      ADD TO STMNT NORM TO
372         STO      DT104 1   COMPUTE POINTER TO NXT STMN
373   *
374         LD       STID      TEST IF END-STMNT
375         S      3 H1000-Z   END STMNT ID CONSTANT
376         BSC  L   F3011, -  BR IF END-STMNT
377         S        DTAID     DATA STATEMENT TEST CONSTAN
378         BSC  L   DT201, -  BRANCH IF DATA STATEMENT
379   DT104 LDX  L1  *-*       MOVE POINTER TO NEXT STMNT
380         MDX      DT102     CONTINUE SCAN FOR DATA STMN
381   *
382   *                CONSTANTS, TEMP STORAGE
383   *
384   TMP1  DC         0       POINTER TO CURRENT STMNT
385   STID  DC         0       STATEMENT ID TYPE
386   DPLF  DC         0       DUPLICATION FACTOR
387   DTAID DC         /F800-/1000
388   *
389   *                MOVE POINTER TO NEXT DATAGROUP
390   *
391   DT201 MDX    1 1
392   *
393         SLA        0       NOP
394         STX    1 TMP1      SAVE CURRENT STRING POINTER
395         LD         TMP1    COMPARE CURRENT STRING LOC
396         S          DT104 1 WITH START OF NEXT STMNT
397         BSC  L     DT104,- BR IF END OF STATEMENT
398   *
399   *                SAVE ADDR OF CONSTANT STRING - 1
400   *                MOVE POINTER TO FIRST VARIABLE
401   *
402         STX  L1    ADCST   SAVE ADDRESS
403         LD     1 0         GET LENGTH OF CON STRING WD
404         AND    3 H0007-Z   MASK OUT ALL BUT LENGTH
405         STO        DT202 1 SAVE LENGTH
406         LD     1 0         GET WD HAVING DUPCTN FACTOR
407         SRA        3       RIGHT JUSTIFY FACTOR IN ACC
408         STO        DPLF    STORE DUPLICATION FACTOR
409         MDX    1 1         MOVE I/P PT PAST DATA HEADE
410   DT202 MDX  L1  *-*       MOVE PT TO FIRST VAR NAME
411   DT203 LD     1 0         GET DATA POINTER
412         BSI    3 GETST-Z   LOCATE SYMBOL TABLE ID WORD
413         LD   I   GET4      GET SYMBOL TABLE ID WORD
414         AND    3 H1800-Z   MASK ID WD FOR DIMENSION BI
415         BSC  L   DT301,Z   BR IF DIMENSIONED VARIABLE
416   *
417   *                OUTPUT DATA FOR A SINGLE VARIABLE
418   *
419         LD   I   GET1 1    GET LOCATION
```

```
420  DT211 BSI      DHT       OUTPUT DATAHDR IF NECESSARY
421        BSI      DTAC      OUTPUT DATA CONSTANT
422  DT221 MDX  L   DPLF,-1   DECR DUPLICATION FACTOR
423        MDX      DT222     BR IF NOT END OF VARIABLES
424        MDX      DT201     BR IF END OF DTAGROUP
425  DT222 MDX    1 1         INCREMENT POINTER
426        MDX      DT203     BR TO PROCESS NEXT VARIABLE
427  *
428  *              DIMENSIONED VARIABLE ENCOUNTERED
429  *
430  DT301 EQU      *         ENTRY POINT LABEL
431        LD       DPLF       LD DUP FACTOR FROM DAT V1M
432        STO    3 ADUPF-Z    *AND PUT IT IN TEMP ON V1M
433        SLA      16        CLEAR ACC
434        STO    3 OFSET-Z   CLEAR OFFSET DISPLACEMENT W
435  *
436        LD     1 0         LOAD DATA POINTER WORD
437        SLA      1         SHIFT DISPLACEMENT INDR BIT
438        BSC  L   DT320,-    BRANCH, NO OFFSET
439  *
440        MDX    1 1         POSITION POINTER
441        LD     1 0         GET DISPLACEMENT WORD
442        STO    3 OFSET-Z     SAVE THE OFF SET VALU V1M
443        LD     1 -1         LD DATA POINTER WORD   V1M
444        SLA      2          PUT BIT 1 IN SIGN POSI V1M
445        BSC  L   DT320,-    BR IF WD NOT SYM TBL P V1M
446        LD     3 ONE-Z      ELSE GET A ONE         V1M
447        STO    3 ADUPF-Z    *AND PUT IT IN TEMP DU V1M
448  *
449  DT320 EQU      *         ENTRY POINT LABEL
450        LD   L   CCWD      VSIZE SIZE OF REAL VARIABLE
451        SLA      14        PLACE INDICATOR IN CARRY
452        LD     3 TWO-Z     STANDARD VARIABLE SIZE
453        BSC      C         SKIP IF STANDARD PRECISION
454        LD     3 THREE-Z   EXTENDED VARIABLE SIZE
455        STO    3 VSIZE-Z   SAVE VARIABLE SIZE
456  *
457        LD   I   GET4      LOAD SYMBOL TABLE ID WORD
458        SLA      1         TEST BIT 1  REAL OR INTEGER
459        BSC  L   DT340,-   BRANCH IF REAL VARIABLE
460  *
461        LD   L   CCWD      VSIZE   SIZE OF INTEGER VAR
462        SLA      10        PLACE INDICATOR IN CARRY
463        LD     3 ONE-Z     ONE WORD INTEGER SIZE
464        BSC      C         SKIP ON NO ONE WORD INTEGER
465        STO    3 VSIZE-Z   SAVE VARIABLE SIZE
466  *
467  DT340 EQU      *         ENTRY POINT LABEL
468  *
469        LD     3 GET4-Z    COMPUTE ARRAY SIZE IN WORDS
470        S      3 THREE-Z   FIRST COMPUTE ADDR OF FIRST
471        STO      * 1       WORD OF ARRAY SIZE
472        LD   L   ***       LOAD ARRAY SIZE FROM SYM TB
473        M      3 VSIZE-Z   MULTIPLY BY VARIABLE SIZE
474        SLT      16        PUT PRODUCT IN ACC
475        STO    3 ASIZE-Z   SAVE IN ARRAY SIZE
476  *
477        S      3 OFSET-Z   COMPUTE NO.OF ELEMENTS THAT
478        SRT      16        CAN BE FILLED.
479        D      3 VSIZE-Z   DIVIDE BY VARIABLE SIZE
```

```
480          STO    3 NOELM-Z  SAVE NO, ELEMENTS TO FILL
481   *
482          S      3 ADUPF-Z   SUB TEMP DUP FACTOR     V1M
483          BSC  L   DT330,    BR, FILL ARRAY IF NOELM LE
484   *                         DUPLICATION FACTOR
485          LD     3 ADUPF-Z   GET TEMP DUP FACTOR     V1M
486          STO    3 NOELM-Z
487   *
488          M      3 VSIZE-Z   COMPUTE ARRAY SIZE
489          SLT      16        NO, ELEMENTS * VAR SIZE
490          A      3 OFSET-Z   PLUS OFFSET
491          STO    3 ASIZE-Z   ARRAY SIZE
492   *
493   DT330  EQU      *         ENTRY POINT
494          LD       DPLF      ADJUST DUPLICATION FACTOR
495          S      3 NOELM-Z   SUBTRACT NO, ELEMENTS
496          A      3 ONE-Z     PLUS ONE
497          STO      DPLF      SAVE NEW DUPLICATION FACTOR
498   *
499          LD     3 NOELM-Z   GET NO, ELEMENTS TO FILL
500          SRA      1         REMOVE LOW ORDER BIT
501          BSC  L   DT350, -  BRANCH IF REMAINDER   0
502   *
503          LD   I   GET4      LOAD SYMBOL TABLE ID WORD
504          SLA      1         TEST FOR INTEGER VARIABLE
505          BSC  L   DT350,-   BRANCH, NOT INTEGER VARIABL
506   *
507          LD   L   CCWD      TEST CONTROL CARD WORD
508          SLA      9         FOR ONE WORD INTEGERS
509          BSC  L   DT350, Z  BRANCH, ONE WORD INTEGERS
510   *
511          SLA      5         EXTENDED PREC BIT TO CARRY
512          LD     3 NOELM-Z   LOAD NO, ELEMENTS
513          BSC      C         TEST CARRY FOR EXTENDED PRE
514          SLA      1         EXTENDED PREC MPY NO, ELTS*
515          A      3 NOELM-Z   ADD ACTUAL NO, ELEMENTS
516          STO    3 NOELM-Z   NO,ELEMENTS *2 OR 3 - SAVE
517   *
518   DT350  EQU      *         ENTRY POINT LABEL
519          LD   I   GET1 1    PLACE DSF HEADER
520          S      3 ASIZE-Z   DATA WD ADDR - ARRAY SIZE
521          A      3 VSIZE-Z   PLUS VAR SIZE
522          BSI      DHT       BR TO SEE IF DATA HDR O/P
523   *
524   DT360  LD     3 NOELM-Z   SET DATA GROUP OUTPUT
525          BSI      DTAC      OUTPUT CONSTANTS
526          MDX  L   NOELM,-1  DECR COUNT ON NO, ELEMENTS
527          MDX      DT360     CONTINUE IF MORE O/P
528          MDX      DT221     FINISHED, RETURN
529   *
530   *                SUBROUTINE
531   *                TEST IF DATAHEADER NECESSARY
532   *                ACC CONTAINS ADDR OF VAR TO BE O/P
533   *                OUTPUT DATA HEADER IF NECESSARY
534   *
535   DHT    DC        0         LINK ENTRY POINT
536          S    L   LOCTR      TEST LOCATION COUNTER
537          BSC  L   DHT2, -   BR IF LOC   LOCTR, NO HDR
538   *                         IS NEEDED
539          A    L   LOCTR     RESET LOCATION COUNTER
```

```
540  DHT1  STO  L  LOCTR     SAVE
541        BSI  L  WRITE      WRITE RECORD
542  DHT2  BSC  I  DHT       RETURN
543  *
544  *               SUBROUTINE OUTPUT DATA-CONSTANT
545  *
546  *               FROM THE STRING
547  DTAC  DC        0         ENTRY POINT
548        LD     3  ADCST-Z   INSERT ADDRESS OF CON STRIN
549        STO       DTAC2 1   SAVE ADDRESS OF CON STRING
550        LD     L  DT202 1   INITLZ CONSTANT STRING COUN
551        STO    3  CLCT-Z    SAVE CONSTANT STRING COUNT
552  DTAC1 MDX    L  DTAC2 1,1 INCR ADDR OF CON STRING
553  DTAC2 LD     L  *-*       GET CONSTANT FR STRING
554        BSI    3  OUABS-Z   OUTPUT
555        MDX    L  CLCT,-1   CONSTANT STRING COUNT
556        MDX       DTAC1     BR IF PART OF CNST REMAINS
557        BSC    I  DTAC      RETURN IF END OF CONSTANT
558  *
559  **
560  *
561  *
562  *               INITIALIZE LOCATIONCOUNTER
563  F3011 LD     L  SOFXT       GET WORK AREA SIZE      V1M
564  *                        (THIS INCLUDES DEF. FILE WORDS
565        BSI       DHT         TEST FOR NEW CARD
566  *
567  *
568  *               OUTPUT CONSTANTS
569  *               SCANS SYMBOL TABLE AND OUTPUTS REAL
570  *               CONSTANTS, FOLLOWED BY A SCAN TO
571  *               OUTPUT INTEGER CONSTANTS
572  *
573  *               INITIALIZE SYMBOL TABLE FOR OUTPUT
574  *               OF CONSTANTS
575  *
576  INIST LD     L  SOFST     LOAD START LOC OF SYMBOL TB
577        STO    3  SYMTP-Z   SYMBOL TABLE POINTER
578  *
579  *               TEST IF END OF SYM TABLE
580  Q1012 LD       3  SYMTP-Z   SYMBOL TABLE POINTER
581        S        L  EOFST     END OF SYMBOL TABLE
582        BSC      L  Q1041,    BR IF END OF SYMBOL TABLE
583  *
584        LD       I  SYMTP     LOAD ID WORD FROM SYMBOL TB
585        BSC         Z         SKIP IF NOT ENTRY OF CON
586        MDX         Q1021     BR IF ENTRY CONSTANT
587  *
588  *               MOVE SYMBOL TABLE POINTER
589  *               BY 3 WORDS FOR MOST SYMBOLS
590  *               OR 6 WORDS FOR DIMENSIONED VARIABLES
591  Q1013 LD       I  SYMTP     LOAD SYMBOL TABLE ID WORD
592        AND      3  H1800-Z   TEST FOR DIMENSIONED VAR.
593        BSC         Z         SKIP IF NOT DIMENSIONED VA
594        LD       3  HFFFD-Z   INCR BY -3 WDS FOR DIM VAR
595        A        3  HFFFD-Z   INCR BY -3 WDS FOR ALL ENT
596        A        3  SYMTP-Z   ADD TO SYMBOL TABLE POINTE
597        STO      3  SYMTP-Z   SAVE IN SYMBOL TABLE POINT
598        MDX         Q1012     CONTINUE SCAN OF SYMBOL TB
599  *
```

```
600   *              CONSTANT FOUND IN SYMBOL TABLE
601   *              OUTPUT REAL AND INTEGER CONSTANTS IN
602   *              ABSOLUTE MODE
603   *
604   Q1021 LDX   I1 SYMTP     PUT SYMBOL TABLE POINTER X
605   *
606   *              SAVE LOCATION COUNTER
607   Q1022 LD    L  LOCTR     LOAD LOCATION COUNTER
608         STO    3 TEMP2-Z   SAVE IN TEMPORARY STORAGE
609   *
610         LD     1 0         GET SYMBOL TABLE CONSTANT
611         SLA      1         SHIFT INTEGER BIT TO SIGN
612         BSC      -         SKIP IF INTEGER CONSTANT
613         MDX      Q1031     BR IF NOT
614   *
615   *              TEST IF INTEGER CONSTANT OUTPUT PASS
616         LD     3 COSW-Z    LOAD INTEGER CON O/P SWITC
617         BSC  L Q1013, -    BR IF NOT INTEGER CON O/P
618   *
619         LD     1 1         SECOND WORD IN SYMBOL TABL
620   *                        ENTRY TO ACCUMULATOR
621         MDX      Q1033     BR TO O/P IN ABSOLUTE MODE
622   *
623   *              TEST IF REAL CONSTANT PASS
624   Q1031 LD     3 COSW-Z    LOAD INTEGER CON O/P SWITC
625         BSC  L Q1013,Z     BR IF INTEGER CON INDICATED
626   *
627   *              TEST IF EXTENDED PRECISION
628         LD   L CCWD        LOAD CONTROL CARD WORD
629         SLA      13        SHIFT EXTENDED PREC BIT
630         BSC      -         SKIP IF EXTENDED PREC
631         MDX      Q1032     BR IF NOT EXTENDED PREC
632   *
633         LD     1 0         LOAD SYMBOL TABLE ID WORD
634         AND    3 H00FF-Z   MASK ALL BUT BITS 8-15
635         BSI    3 OUABS-Z   SUBR OUTPUT, ENTRY   ABS
636   Q1032 LD     1 1         SECOND WORD IN ENTRY TO ACC
637         BSI    3 OUABS-Z   SUBR OUTPUT, ENTRY   ABS
638         LD     1 2         THIRD WORD IN SYMBOL TABLE
639   *                        ENTRY TO ACC
640   Q1033 BSI    3 OUABS-Z   SUBR OUTPUT, ENTRY   ABS
641   *
642   *              INSERT ALLOCATION INTO SYM,TBL,
643         LD     3 TEMP2-Z   GET ALLOC CON FROM TEMP STO
644         STO    1 1         STORE IN 2ND WD OF SYM TBL
645   *
646         MDX      Q1013     BR TO CONTINUE SYM TBL SCAN
647   *
648   *              TEST IF THAT WAS THE SECOND
649   *                INTEGER CONSTANT  PASS
650   *
651   Q1041 LD     3 COSW-Z    CONSTANT OUTPUT SWITCH
652         BSC  L Q1051,Z     BR IF SECOND PASS
653         MDX  L COSW,1      CHANGE CONSTANT O/P SWITCH
654         MDX      INIST     GO TO NEXT PASS
655   *
656   *              OUTPUT OF CONSTANTS COMPLETED
657   *              START OUTPUT  OF STRING DATA
658   *
659   *              TEST THAT OUTPUT LOCATION COUNTER
```

```
660   *               CHECKS WITH THE ONE ASSUMED AT
661   *               STATEMENT ALLOCATION. IF NOT,
662   *               OUTPUT DUMMY WORD.
663   Q1051 LD    L   SOFXT     LOAD SIZE OF WORK AREA
664         A     L   SOFGT     PLUS SIZE OF CONSTANT AREA
665         S     L   LOCTR     LESS THE LOCATION COUNTER
666         BSC   L   EXIT,     BR IF SIZE WK AREA   CONST
667   *               AREA L.E. LOCATION COUNTER
668         SLA       16        CLEAR ACC
669         BSI     3 OUABS-Z   OUTPUT DUMMY WORD
670         MDX       Q1051     CONTINUE LOOP
671   *               BR TO LOADING OF NEXT PHASE
672         BSC   L   EXIT
673   *
674   *               PROGRAM BEING COMPILED IS A SUBR
675   *               INITIALIZE SUBR, OUTPUT
676   *
677   INISR LD      3 H0401-Z           SET SUBPROGRAM TYPE
678         BSC   L   Q101X     RETURN TO CONTINUE PROCESSI
679   *
680   * WRITE BINARY DATA
681   *
682   WRITE DC        *-*               ENTRY POINT
683         LDX     3 54
684         SLA       17                CLEAR CARRY
685         LD    L   RECCT             RECORD COUNT
686   WR01  A     L3  BUFF              FORM CHECKSUM
687         BSC       C
688         A         ONE
689         MDX     3 -1
690         MDX       WR01              LOOP
691         RCP       4,4
692         RIC       4,4         MAKE 2*S COMPLIMENT
693         STO   L   WD2               CHECKSUM
694         BSI   L   LIO               WRITE BINARY RECORD
695         DC        /2104
696         DC        BUFF
697         DC        0
698         BSI   L   LIO               WAIT FOR TRANSFER
699         DC        /F004
700         LDX     3 53
701         SLA       17
702   WR02  STO   L3  WD1               CLEAR BUFFER
703         MDX     3 -1
704         MDX       WR02              LOOP
705         LDX   L3  Z                 RESTORE Z POINTER
706         LD      3 CD16-Z            -16
707         STO   L   INCT              CNT,-BIT COUNT = 06
708         LDD       WR03
709         STD   L   BUFPT             BUFPT=WD10,INPT=WD4
710         LD    L   LOCTR
711         STO   L   WD1               NEW LOCATION COUNTER
712         LD        WR03+2            /A00
713         STO   L   WD3               DATA RECORD TYPE
714         MDX   L   RECCT,1     INCREMENT RECORD COUNT
715         BSC   I   WRITE       EXIT
716         BSS   E   0
717   WR03  DC        WD10              DATA START
718         DC        WD4               CONTROL START
719         DC        /A00
```

```
720   *
721           BSS  E  0
722   STONA   DC      0          STORED NAME WD 1
723           DC      0          STORED NAME WD 2
724   *
725   ZERO    DC      0          CONSTANT
726   Z       EQU     ZERO       CONSTANT
727   ONE     DC      1          CONSTANT
728   TWO     DC      2          CONSTANT
729   THREE   DC      3          CONSTANT
730   FOUR    DC      4          CONSTANT
731   SIX     DC      6            CONSTANT
732   H0002   DC      /0002      CONSTANT
733   H0003   DC      /0003      CONSTANT
734   H0005   DC      /0005      CONSTANT
735   SEVEN   DC      7          CONSTANT
736   H0200   DC      /0200      CONSTANT
737   H0201   DC      /0201      MAINL REL TYPE W PREC CONST
738   H0401   DC      /0401      CONSTANT
739   C320    DC      320        CONSTANT
740   H1800   DC      /1800      CONSTANT
741   HFFFD   DC      /FFFD      CONSTANT
742   HFEFF   DC      /FEFF      EXT PRECISION MASK
743   H0007   DC      /0007      CONSTANT
744   H00FF   DC      /00FF      CONSTANT
745   CD16    DC      16         CONSTANT
746   H07FC   DC      /07FC      CONSTANT
747   HF800   DC      /F800      CONSTANT
748   HF000   DC      /F000      ID FOR  DEFINE FILE
749   H07FF   DC      /07FF      CONSTANT
750   H1000   DC      /1000      CONSTANT
751   H2000   DC      /2000      CONSTANT
752   H8000   DC      /8000      MASK TO SEE IF DISK IO BITO
753   H4000   DC      /4000      CONSTANT
754   BFEND   DC      BUFF+55          BUFFER END
755   OFSET   DC      *-*        DISPLACEMENT WD FR DATA PT
756   VSIZE   DC      *-*        VARIABLE SIZE   1,2 OR 3
757   ASIZE   DC      *-*        ARRAY SIZE DATA STMNT
758   NOELM   DC      *-*        NO. ELEMENTS TO BE FILLED
759   ADCST   DC      *-*        ADDR CONSTANT STRING - 1
760   CLCT    DC      *-*        LENGTH OF CONSTANT STRING
761   HOSW    DC      *-*        DATA GROUP O/P SWITCH
762   *
763   NORM    DC      0          WD COUNT I/P STRING
764   COSW    DC      0          CONSTANT PASS SWITCH
765   TEMP2   DC      0          TEMPORARY STORAGE
766   SYMTP   DC      0          SYMBOL TABLE POINTER
767   SCT     DC      0          SEVEN COUNT
768   ADUPF   DC      *-*        TEMPORARY DUPLICATION FACTO
769   *
770   *
771   *               SUBROUTINE GETST
772   *               GET OBJECT TIME ADDRESS OR
773   *               ALPHABETIC NAME FROM SYMBOL TABLE
774   *               ON ENTRY ACC   STRING POINTER TO
775   *               SYMBOL TABLE
776   *
777   GETST   DC      0          LINK ENTRY POINT
778           AND   3 H07FF-Z    MASK OUT ALL BUT VAR PT
779           STO     GET1 1     SAVE VARIABLE POINTER
```

```
780          LD    L  SOFST     LOAD START OF SYMBOL TABLE
781          S        GET1 1    SUBTRACT POINTER IN SYM TBL
782          S        GET1 1    *THREE
783          S        GET1 1    *TIMES
784          A      3 H0005-Z   ADD 5 TO POS AT END OF DATA
785          STO      GET1 1    SAVE POINTER
786          S      3 TWO-Z     MOVE PT BACK TO ADDR OF ID
787          STO      GET4      STORE ADDR OF ID-WD
788   GET1   LD    L  *-*       LOAD VARIABLE FROM SYM TBL
789          RTE      16        PUT IN EXTENSION
790          MDX   L  GET1 1,-1 DECR POINTER
791          SLA      0         GUARDS AGAINST SKIP
792          LD    I  GET1 1    GET DATA WD FR SYM TBL
793          BSC   I  GETST     RETURN
794   GET4   DC       *-*       ADDR OF SYM TBL ID WORD
795   *
796   *               OUTPUT ADDRESS OR CONSTANT IN
797   *               ABSOLUTE MODE
798   *               ACCUMULATOR AND EXTENSION CONTAIN
799   *               WORDS TO BE OUTPUT
800   *
801   *               ENTRY   ABSOLUTE OUTPUT
802   OUABS  DC       0         LINK
803   T8011  STD    3 STONA-Z   STORE ACC AND EXTENSION
804          LD     3 ZERO-Z    ZERO-ZERO
805          BSI      IBITS     TO INDICATOR AREA
806   *
807   T8012  LD     3 STONA-Z   GET STORED ACCUMULATOR
808          BSI      TOBUF     MOVE TO BUFFER
809   *
810   T8016  BSC    I  OUABS    RETURN
811   *
812   *               OUTPUT ADDRESS OR CONSTANT IN
813   *               RELOCATABLE MODE
814   *
815   *               ENTRY   RELATIVE OUTPUT
816   *               RELATIVE OUTPUT  INCLUDES TEST
817   *               WHETHER VARIABLE IS IN COMMON, IN
818   *               WHICH CASE OUTPUT IS ABSOLUTE
819   *
820   OUREL  DC       0         LINK ENTRY POINT
821   T8013  STD    3 STONA-Z   STORE ACC AND EXTENSION
822          LD       OUREL     SET LINK RETURN ADDRESS
823          STO      OUABS     IN ABSOLUTE LINK EXIT POINT
824          LD    L  GET1 1    LOAD ADDR OF SYM TBL DATA W
825          S      3 ONE-Z     SUBTRACT.1 TO GET ADDR ID W
826          STO      T8014 1   INSERT ADDR OF SYM TBL ID W
827   T8014  LD    L  *-*       LOAD SYM TBL ID WORD
828          AND    3 H2000-Z   TEST FOR COMMON INDR BIT
829          BSC   L  T8015, -  BR IF NOT IN COMMON
830          MDX   L  LOCTR,-1   FIX FOR S5232            V1M
831          NOP                 TAKE NO CHANCES          V1M
832          LD     3 H0003-Z   INDICATOR BIT MASK
833          BSI      IBITS     PUT FIRST 2 INDR BITS IN
834          LD     3 STONA-Z   INDICATOR POINTER
835          BSC      -         SKIP IF BLANK COMMON
836          S      3 H4000-Z   ADJ ADDR OF LABELED COMMON
837          BSI      TOBUF     O/P ADDR TO BUFFER AREA
838          LD     3 STONA-Z   GET ADDR TO BUFFER
839          SLA      1         PUT BLANK COMMON BIT TO C
```

```
840          LD     3 H0002-Z   SET INDR FOR INSKEL COMMON
841          BSC     C          SKIP UNLESS BLANK COMMON
842          LD     3 H0003-Z   SET BLANK COMMON INDR
843          BSI     IBITS      PUT OUT INDICATOR BITS
844          LD     3 STONA-Z   GET ADDRESS
845          BSC     -          SKIP IF BLANK COMMON
846          S      3 H4000-Z   ADJ LABELED COMMON ADDR
847          BSI     TOBUF      O/P ADDRESS TO BUFFER
848          MDX     T8016      RETURN
849   *
850   T8015  LD     3 ONE-Z     ZERO-ONE FOR INDICATOR BITS
851          BSI     IBITS      MOVE TO INDICATOR AREA
852          MDX     T8012      RETURN
853   *
854   *
855   *               INDICATOR BITS
856   *               ACCUMULATOR CONTAINS BIT PATTERNS TO
857   *               MASK INTO THE INDICATOR POINTER
858   *               BASED ON SHIFTS SPECIFIED IN THE
859   *               INDICATOR COUNTER
860   *
861   *
862   IBITS  DC       0          LINK ENTRY POINT
863          STO      IBIT5      SAVE BIT PATTERNS
864          LD     L INCT       PRODUCE SLA AND SRA
865          A        IBIT6      INSTRUCTIONS AND INSERT
866          STO      IBIT3      LENGTH OF SHIFT DEPENDS ON
867          A        IBIT7      INDICATOR COUNTER
868          STO      IBIT2      SET UP SRA INSTRUCTION
869          LD     I INPT
870   IBIT2  SRA      *-*        SRA  INCT - 2
871          A        IBIT5      TEMP VALUE FROM ACC
872   IBIT3  SLA      *-*        SLA  INCT - 2
873          STO    I INPT
874          MDX    L INCT,-2    DECR INDICATOR COUNTER
875          MDX      IBIT4      BR IF NOT ZERO   NO SKIP
876          MDX    L INPT,1          INCR, INDICATOR PTR,
877          LD     3 CD16-Z     REINITIALIZE INDICATOR CTR
878          STO    L INCT       TO 16
879   IBIT4  BSC    I IBITS      RETURN
880   IBIT5  DC       0          TEMPORARY STORAGE
881   IBIT6  DC       /0FFE      ,SLA 0, - 0002
882   IBIT7  DC       /0800      ,SRA 0, - ,SLA 0,
883   *
884   *               SUBROUTINE
885   *               MOVE TO BUFFER
886   *               ACCUMULATOR CONTAINS INDICATOR
887   *               POINTER
888   *
889   TOBUF  DC       0          LINK ENTRY POINT
890          STO    I BUFPT      SAVE WORD IN BUFFER
891          MDX    L LOCTR,1    INCR LOCATION COUNTER
892          MDX    L BUFPT,1     INC, BUFFER POINTER
893          MDX    L WD3,1           INCREMENT DATA COUNT
894   *               TEST IF END OF BUFFER
895          LD     L BUFPT
896          S      3 BFEND-Z
897          BSC      -                SKIP IF NOT FULL
898          BSI    3 WRITE-Z
899          BSC    I TOBUF      REUTRN
```

```
900   *
901   *              PROGRAM ENTRY
902   *              TEST IF OUTPUT IS TO BE DEFERRED
903   *
904   ENT    LD   L  ERROR
905          BSC  L  START, -- BRANCH TO START IF NO
906   *                          ERROR
907   *
908   *              LOAD NEXT PHASE
909   *
910   EXIT   BSI  L  ROLRX      CALL DOWN PHASE 27
911          DC      27                NEXT PHASE NUMBER
912   *
913          BSS     OVERL-**320*3  PHASE-26 PATCH AREA
914          END     NEQ
```