```
TTTTTTTTTT   EEEEEEEEEE     SSSSSSSS   TTTTTTTTTT
TTTTTTTTTT   EEEEEEEEEE     SSSSSSSS   TTTTTTTTTT
    TT       EE           SS              TT
    TT       EE           SS              TT
    TT       EE           SS              TT
    TT       EE           SS              TT
    TT       EEEEEEEE       SSSSSS        TT
    TT       EEEEEEEE       SSSSSS        TT
    TT       EE                   SS      TT
    TT       EE                   SS      TT
    TT       EE                   SS      TT          ....
    TT       EE                   SS      TT          ....
    TT       EEEEEEEEEE   SSSSSSSS        TT          ....
    TT       EEEEEEEEEE   SSSSSSSS        TT          ....

LL             SSSSSSSS   TTTTTTTTTT       ;;;;         11
LL             SSSSSSSS   TTTTTTTTTT       ;;;;         11
LL           SS               TT          ;;;;       1111
LL           SS               TT          ;;;;       1111
LL           SS               TT                       11
LL           SS               TT                       11
LL             SSSSSS         TT          ;;;;         11
LL             SSSSSS         TT          ;;;;         11
LL                   SS       TT          ;;;;         11
LL                   SS       TT          ;;;;         11
LL                   SS       TT            ;;         11
LL                   SS       TT            ;;         11
LLLLLLLLLL   SSSSSSSS         TT          ;;         111111
LLLLLLLLLL   SSSSSSSS         TT          ;;         111111
```

```
1   ********************************************************************************
2   ********************************************************************************
3   ********************************************************************************
4   ***                                                                          ***
5   ***                                                                          ***
6   ***                                                                          ***
7   ***            TEST MODULE FOR  ISI  PRINTERS COMPATIBLE                     ***
8   ***                  with  IBM  3274/6 PROTOCOL                              ***
9   ***                                                                          ***
10  ***                                                                          ***
11  ***                  INTERFACE SYSTEMS, INC.                                 ***
12  ***                  ANN ARBOR, MICHIGAN  48103                              ***
13  ***                                                                          ***
14  ***                                                                          ***
15  ***                                                                          ***
16  ***            FOR MOTOROLA  6809  OR COMPATIBLE MICROPROCESSOR              ***
17  ***                  USING A GENERATION III INTERFACE                        ***
18  ***                                                                          ***
19  ***                                                                          ***
20  ***                                                                          ***
21  ********************************************************************************
22  ********************************************************************************
23  ********************************************************************************
24  *
25  *
26  *              BY:     INTERFACE SYSTEMS, INCORPORATED
27  *                      5855 INTERFACE DRIVE
28  *                      ANN ARBOR, MICHIGAN  48103
29  *
30  *                      RICHARD L. COLE
31  *
32  *                      COPYRIGHT 1982, 1983
33  *                              BY
34  *                      INTERFACE SYSTEMS, INC.
35  *
36  *                      ALL RIGHTS RESERVED
37  *
38  *                      Modified June, Aug 1983
39  *                              inclusion of beg/end print column
40  *
41  *
42  ********************************************************************************
43  ********************************************************************************
44  *
45  *
46          NAM     TEST MODULE
47  *
48  *
49          EXTERN  BUSY            printer BUSY flag
50          EXTERN  DEVMDN          printer model number
51          EXTERN  ENAFLG          printer ENABLED flag (coax-protocol)
52          EXTERN  EXSTAT          "special status"
```

```
                                53          EXTERN  LSTCOM          last coax command (bottom board only)
                                54          EXTERN  LSTORD          last ORDER received
                                55          EXTERN  PIDENT          terminal id field
                                56          EXTERN  SWFLAG          front parel switch pressed flag
                                57          EXTERN  SWVEC           base of switch vector
                                58          EXTERN  VERSIO          software version number
                                59  *
                                60  *
                                61          INTERN  AUTDMP
                                62          INTERN  CONCHK          return continue/cancel flag for multi-line test*
                                63          INTERN  TEST
                                64          INTERN  TSTBUF          test print buffer
                                65  *
                                66  *
                                67          INCLUD  DK3:[140,2]RANGE.ACT
                                68  *
=2000                           69  .RAMB.  EQU     $2000           target/target variable RAM
=27FF                           70  .RAME.  EQU     $27FF
=C000                           71  .ROMB.  EQU     $C000           target program base address
=FFFF                           72  .ROME.  EQU     $FFFF           target program limit
                                73  *
                                74          INCLUD  DK3:[140,2]CALL.MAC
                                75  *       'CALL' MACRO DEFINED
                               134  *
                               135  *
=C000 =FFFF                    136          SECT    TSTROM,REL,RANGE=.ROMB.:.ROME.,ALIGN=PAGE
=2000 =27FF                    137          SECT    TSTRAM,REL,RANGE=.RAMB.:.RAME.
                               138  *
                               139  *
=FFFF                          140  TRUE.   EQU     -1              boolean "true"
=0000                          141  FALSE.  EQU     0               boolean "false"
=0008                          142  ADCOD   EQU     8               test switch code for auto-hex-dump
=0010                          143  DBC_SP  EQU     $10             DBC code for (space)
=0020                          144  DBC_0   EQU     $20             DBC graphic 0
=0029                          145  DBC_9   EQU     $29             DBC graphic 9
=00A2                          146  DBC_C   EQU     $A2             DBC graphic C
=00AA                          147  DBC_K   EQU     $AA             DBC graphic K
=00B7                          148  DBC_X   EQU     $B7             DBC graphic X
                               149  *
                               150  ********************************************************************************
```

```
                                152  ********************************************************************************
                                153  *
        =0000                   154          RSECT   TSTRAM
                                155  *
0000'   =0001                   156  GDUMP   RMB     1               "good dump" (not aborted) flag
0001'   =0002                   157  LSTADR  RMB     2               address beyond end of dump
        =0010                   158          RMB     16              pre-buffer space
0013'   =0050                   159  TSTBUF  RMB     80              test print buffer
                                160  *
                                161  *
        =0000                   162          RSECT   TSTROM
                                163  *
0000'   BD 0015'                164  TEST    JSR     TSWIT           get, mask & shift test switch code
0003'   48                      165          LSLA                    double for index
0004'   8E 001D'                166          LDX     #TSTTAB         base of table
0007'   AD 96                   167          JSR     [A,X]           go to appropriate subroutine
0009'   39                      168          RTS
                                169  *
000A'   BD 0015'                170  AUTDMP  JSR     TSWIT           get test switches
000D'   81 08                   171          CMPA    #ADCOD          is auto dump specified?
000F'   26 03                   172          BNE     99$             branch if no
0011'   BD 008F'                173          JSR     FDUMP           yes, do full dump
0014'   39                      174  99$     RTS
                                175  *
                                176  *
                                177  *       get, mask and shift test switch code
                                178  *
0015'   B6 0000"                179  TSWIT   LDA     SWVEC+1         get current code
0018'   44                      180          LSRA                    shift right 4 places (masking in process)
0019'   44                      181          LSRA
001A'   44                      182          LSRA
001B'   44                      183          LSRA
001C'   39                      184          RTS
                                185  *
                                186  *       test decode table
                                187  *
001D'   0037'                   188  TSTTAB  FDB     VERS            0 - user version print
001F'   0037'                   189          FDB     VERS            1 - print version, etc.
0021'   0088'                   190          FDB     STAT            2 - print "special status"
0023'   00BF'                   191          FDB     FDUMP           3 - dump all
0025'   00E6'                   192          FDB     SDUMP           4 - dump stack/var ram only
0027'   0032'                   193          FDB     DVTST           5 - device dependant test
0029'   0036'                   194          FDB     MEMTST          6 - do a memory check ??????????????????
0028'   0031'                   195          FDB     NULL            7 - normal, no test
002D'   0031'                   196          FDB     NULL            8 - auto-dump, no test
002F'   0171'                   197          FDB     TPRNT           9 - Infinite test print
                                198  *
                                199  ********************************************************************************
```

```
                        201   ********************************************************************************
                        202   *
                        203   *       no test
                        204   *
                        205   *
0031'  39               206   NULL    RTS
                        207   *
                        208   ********************************************************************************
                        209   *
                        210   *       device dependent test
                        211   *
                        212+  DVTST   CALL    DEVTST          link to test program in device module
                        217A          EXTERN  DEVTST
0032'  8D 0008*         221A          JSR     DEVTST          (GO TO DEVTST)
0035'  39               241           RTS
                        242   *
                        243   ********************************************************************************
                        244   *
                        245   *       memory test  --  to be defined
                        246   *
0036'  39               247   MEMTST  RTS
                        248   *
                        249   ********************************************************************************
                        250   *
                        251   *       print version, model, checksum (of rom)
                        252   *
0037'  BD 0250'         253   VERS    JSR     SETBUF          set up the test buffer
003A'  BD 025C'         254           JSR     NL              test prints should always start with NL
                        255   *
                        256   *       copy up to 62 characters from "version" message to test buffer
                        257   *
003D'  CE 00CA*         258           LDU     #VERSIO         address containing version
0040'  E6 C0            259           LDB     ,U+             Load version byte count
0042'  27 13            260           BEQ     20$             No version string
                        261   *
0044'  C1 3E            262           CMPB    #62             Version string too long ?
0046'  23 02            263           BLS     10$             No, skip
0048'  C6 3E            264           LDB     #62             Load max. byte ccunt
004A'  34 04            265   10$     PSHS    B               Save byte count
                        266   *
004C'  A6 C0            267   11$     LDA     ,U+             Load code
                        268+          CALL    DEVOUT          Transfer to test print buffer
                        273A          EXTERN  DEVOUT
004E'  8D 000C*         277A          JSR     DEVOUT          (GO TO DEVOUT)
0051'  6A E4            297           DEC     ,S              Last code ?
0053'  26 F7            298           BNE     11$             No, loop
0055'  32 61            299           LEAS    1,S             Clean up stack
                        300   *
0057'  B6 0002*         301   20$     LDA     DEVMDN          output model number
005A'  8D 022F'         302           JSR     HEX
005D'  86 10            303           LDA     #DBC_SP         follow with a space
                        304+          CALL    DEVOUT
```

```
005F'  BD 000C*          313A          JSR     DEVOUT          (GO TO DEVOUT)
0062'  86 A2             333           LDA     #DBC_C          a C
                         334+          CALL    DEVOUT
0064'  BD 000C*          343A          JSR     DEVOUT          (GO TO DEVOUT)
0067'  86 AA             363           LDA     #DBC_K          a K
                         364+          CALL    DEVOUT
0069'  BD 000C*          373A          JSR     DEVOUT          (GO TO DEVOUT)
006C'  86 10             393           LDA     #DBC_SP         another space
                         394+          CALL    DEVOUT
006E'  BD 000C*          403A          JSR     DEVOUT          (GO TO DEVOUT)
0071'  8E C000           423           LDX     #.ROMB.         compute ROM checksum by words
0074'  CC 0000           424           LDD     #0              clear sum
0077'  E3 81             425   30$     ADDD    ,X++            add a word to sum
0079'  8C 0000           426           CMPX    #.ROME.+1       End?
007C'  26 F9             427           BNE     30$
                         428   *
007E'  BD 021D'          429           JSR     OUT4            output checksum
0081'  86 10             430           LDA     #DBC_SP         output a space
                         431+          CALL    DEVOUT
0083'  BD 000C*          440A          JSR     DEVOUT          (GO TO DEVOUT)
0086'  20 06             460           BRA     STAT1           jump into STAT, below
                         461   *
                         462   ********************************************************************************
                         463   *
                         464   *       "special status"        --      Xxxyzab
                         465   *
                         466   *               X is an indicator character
                         467   *               xx represents the "special status" byte
                         468   *               y represents the BUSY flag, F or 0
                         469   *               z represents the ENABLED flag, F or 0
                         470   *               a represents the last ORDER, 0 to 3
                         471   *               b represents the last COMMAND, 0 to 3
                         472   *
0088'  BD 0250'          473   STAT    JSR     SETBUF          set up test buffer
008B'  BD 025C'          474           JSR     NL              begin with a new line
                         475   *
008E'  86 B7             476   STAT1   LDA     #DBC_X          X
                         477+          CALL    DEVOUT
0090'  BD 000C*          486A          JSR     DEVOUT          (GO TO DEVOUT)
0093'  B6 0004*          506           LDA     EXSTAT          the "special status" byte
0096'  BD 0224'          507           JSR     OUT2
0099'  B6 0001*          508           LDA     BUSY            BUSY flag
009C'  BD 022F'          509           JSR     HEX
009F'  B6 0003*          510           LDA     ENAFLG          ENABLED flag
00A2'  BD 022F'          511           JSR     HEX
00A5'  B6 0006*          512           LDA     LSTORD          last ORDER
00A8'  84 03             513           ANDA    #3
00AA'  BD 022F'          514           JSR     HEX
00AD'  B6 0005*          515           LDA     LSTCOM          last COMMAND
00B0'  84 03             516           ANDA    #3
00B2'  BD 022F'          517           JSR     HEX
00B5'  BD 023D'          518           JSR     TSTPUT          Output test print line
```

```
00B3' BD 025C'          519          JSR     NL
00B8' BD 025C'          520          JSR     NL
00BE' 39                521          RTS
                        522  *
                        523  *
                        524  ********************************************************************************
                        525  *
                        526  *       full dump  --  print buffer, eab, stack & var
                        527  *
00BF' BD 0037'          528  FDUMP   JSR     VERS            print version/status as header
00C2' 8E 0000           529          LDX     #$0             dump starting at addr=0
00C5' B6 0000"          530          LDA     PIDENT+2        buffer size (high byte)
00C8' 5F                531          CLRB
00C9' 1F 03             532          TFR     D,U             to buffer_size-1
00CB' BD 00F6'          533          JSR     DUMP            dump ((X)) to ((U)-1)
00CE' 4D                534          TSTA                    was dump aborted?
00CF' 27 21             535          BEQ     DMPEND          branch if yes
                        536  *
                        537  *       eab?
                        538  *
00D1' 7D 0000"          539          TST     PIDENT+1        eab present? (ID byte 1, bit 0)
00D4' 2A 10             540          BPL     SDUMP           branch if no
00D6' 8E 1000           541          LDX     #4096           eab always starts here
00D9' 1F 10             542          TFR     X,D             copy to D
00DB' BB 0000"          543          ADDA    PIDENT+2        calculate end
00DE' 1F 03             544          TFR     D,U
00E0' BD 00F6'          545          JSR     DUMP
00E3' 4D                546          TSTA                    was dump aborted?
00E4' 27 0C             547          BEQ     DMPEND          branch if yes
                        548  *
                        549  *       stack and variable ram  ************************************************
                        550  *
00E6' BD 0037'          551  SDUMP   JSR     VERS            print version/status as header
00E9' 8E 2000           552          LDX     #.RAMB.         beginning
00EC' CE 27FF           553          LDU     #.RAME.         end
00EF' BD 00F6'          554          JSR     DUMP
00F2' BD 025C'          555  DMPEND  JSR     NL              skip a line after dump
00F5' 39                556          RTS                     done
                        557  *
                        558  *
                        559  *       DUMP SUBROUTINE
                        560  *
                        561  *       dumps memory from (X) to (U)-1
                        562  *
                        563  *
00F6' 86 FF             564  DUMP    LDA     #TRUE.          initialize "good (unaborted) dump" flag
00F8' B7 0000'          565          STA     GDUMP
00FB' FF 0001'          566          STU     LSTADR          Save last address
00FE' BD 025C'          567          JSR     NL              ok to skip a line before dumping
                        568  *
                        569  *       dump a line, then check for cancel (TEST switch pressed)
                        570  *
```

```
0101' 108E 0013'          571          LDY   #TSTBUF        Point to test print buffer
0105' BD 0144'            572    10$   JSR   PNTLIN
                          573    *
0108' BD 0267'            574          JSR   CONCHK         check for front panel abort
010B' 27 26               575          BEQ   40$
                          576    *
                          577    *     check for last word dumped.  check for lines to be skipped.
                          578    *
010D' BC 0001'            579          CMPX  LSTADR         Last byte ?
0110' 27 24               580          BEQ   255$           Yes, exit
                          581    *
0112' 1F 13               582          TFR   X,U            starting scan point
0114' A6 1F               583          LDA   -1,X           last byte dumped
                          584    *
                          585    *     scan loop
                          586    *
0116' 1183 0001'          587    20$   CMPU  LSTADR         last line passed?
011A' 27 04               588          BEQ   30$            Yes, print last line
011C' A1 C0               589          CMPA  ,U+            check a byte
011E' 27 F6               590          BEQ   20$            if equal, go look some more
                          591    *
                          592    *     found something to dump  --  should a line be skipped?
                          593    *
0120' 33 5F               594    30$   LEAU  -1,U           back up to data to dump
0122' 1F 30               595          TFR   U,D            compute base address of its line
0124' C4 F0               596          ANDB  #$F0
0126' 34 06               597          PSHS  D
0128' AC E1               598          CMPX  ,S++           same as already pointed to by X?
012A' 27 09               599          BEQ   10$            if yes, go dump it
                          600    *
012C' 1F 01               601          TFR   D,X            no, update X and skip line
012E' BD 025C'            602          JSR   NL
0131' 20 D2               603          BRA   10$
                          604    *
                          605    *     dump aborted
                          606    *
0133' B7 0000'            607    40$   STA   GDUMP          set good dump false
                          608    *
                          609    *     dump is complete
                          610    *
0136' B6 0000'            611    255$  LDA   GDUMP          return no abort/abort flag
0139' 39                  612          RTS
                          613    *
                          614    *     control table
                          615    *
013A' 00 00 00 FF FF 00   616          FCB   FALSE.,FALSE.,FALSE.,TRUE.,TRUE.,FALSE.,FALSE.,FALSE.
0140' 00 00
0142' FF 00               617          FCB   TRUE.,FALSE.
                          618    *
                          619    *     dump line routine  --  display (X), then ((X))...((X+15)), then NL
                          620    *
0144' 1F 10               621    PNTLIN TFR    X,D            print base address
```

```
0146'  BD 021D'          622          JSR     OUT4
0149'  86 10             623          LDA     #DBC_SP         4 spaces
                         624+         CALL    DEVOUT
014B'  BD 000C*          633A         JSR     DEVOUT          (GO TO DEVOUT)
                         653+         CALL    DEVOUT
014E'  BD 000C*          662A         JSR     DEVOUT          (GO TO DEVOUT)
                         682+         CALL    DEVOUT
0151'  BD 000C*          691A         JSR     DEVOUT          (GO TO DEVOUT)
                         711+         CALL    DEVOUT
0154'  BD 000C*          720A         JSR     DEVOUT          (GO TO DEVOUT)
0157'  A6 80             740   10$    LDA     ,X+             get a byte
0159'  BD 0224'          741          JSR     OUT2
015C'  86 10             742          LDA     #DBC_SP         2 spaces
                         743+         CALL    DEVOUT
015E'  BD 000C*          752A         JSR     DEVOUT          (GO TO DEVOUT)
                         772+         CALL    DEVOUT
0161'  BD 000C*          781A         JSR     DEVOUT          (GO TO DEVOUT)
0164'  1F 10             801          TFR     X,D             check for line done
0166'  C4 0F             802          ANDB    #$F
0168'  26 ED             803          BNE     10$             No !
                         804   *
016A'  BD 023D'          805          JSR     TSTPUT          Output test print line
016D'  BD 025C'          806          JSR     NL              new_line
0170'  39                807          RTS
                         808   *
                         809   ********************************************************************************
                         810   *
                         811   *       test print  --  alphabets numbers and special characters
                         812   *
                         813   *               indefinate repeat until cancelled
                         814   *
                         815   *
0171'  BD 025C'          816   TPRNT  JSR     NL              test prints should always start with NL
0174'  108E 01A1'        817   10$    LDY     #TP1            print the test line
                         818+         CALL    DEVPUT
                         823A         EXTERN  DEVPUT
0178'  BD 000D*          827A         JSR     DEVPUT          (GO TO DEVPUT)
                         847+         CALL    DEVCR           return to beginning of line
                         852A         EXTERN  DEVCR
017B'  BD 000E*          856A         JSR     DEVCR           (GO TO DEVCR)
017E'  108E 0202'        876          LDY     #TP2            print underlines
                         877+         CALL    DEVPUT
0182'  BD 000D*          886A         JSR     DEVPUT          (GO TO DEVPUT)
0185'  BD 025C'          906          JSR     NL              new line
0188'  BD 025C'          907          JSR     NL
018B'  BD 0267'          908          JSR     CONCHK          continue?
018E'  26 E4             909          BNE     10$             yes, loop
                         910   *
0190'  39                911          RTS                     no, stop
                         912   *
                         913   *       device coded buffer for test print  ALPHAalphanumbersspecials
                         914   *
```

```
                                    915   *           character density is coded for default  --  10cpi
                                    916   *
0191'  00 00 00 00 00 00            917               FCB     0,0,0,0,0,0,0,0
0197'  00 00
0199'  00 00 00 00 01 51            918               FCB     0,0,0,0,1,81,0,0
019F'  00 00
01A1'  50                           919   TP1         FCB     80              buffer has 80 chars
       =0010                        920               RADIX   16
01A2'  A0 A1 A2 A3 A4 A5            921               FCB     $A0,$A1,$A2,$A3,$A4,$A5,$A6,$A7,$A8,$A9,$AA,$AB,$AC      A-M
01A8'  A6 A7 A8 A9 AA AB
01AE'  AC
01AF'  AD AE AF B0 B1 B2            922               FCB     $AD,$AE,$AF,$B0,$B1,$32,$B3,$B4,$B5,$B6,$B7,$B8,$B9      N-Z
01B5'  B3 B4 B5 B6 B7 B8
01BB'  B9
01BC'  80 81 82 83 84 85            923               FCB     80,81,82,83,84,85,86,87,88,89,8A,8B,8C                    a-m
01C2'  86 87 88 89 8A 8B
01C8'  8C
01C9'  8D 8E 8F 90 91 92            924               FCB     8D,8E,8F,90,91,92,93,94,95,96,97,98,99                    n-z
01CF'  93 94 95 96 97 98
01D5'  99
01D6'  20 21 22 23 24 25            925               FCB     20,21,22,23,24,25,26,27,28,29                             0-9
01DC'  26 27 28 29
01E0'  0D 11 12 13 14 18            926               FCB     0D,11,12,13,14,18,19,1A,2E                          (='"/?!$%
01E6'  19 1A 2E
01E9'  30 31 32 33 34 35            927               FCB     30,31,32,33,34,35,$8E,$8F,0C                        &-.,:+;*)
01EF'  8E 8F 0C
       =000A                        928               RADIX   10
                                    929   *
                                    930   *           underline string
                                    931   *
01F2'  00 00 00 00 00 00            932               FCB     0,0,0,0,0,0,0,0
01F8'  00 00
01FA'  00 00 00 00 01 1B            933               FCB     0,0,0,0,1,27,0,0
0200'  00 00
0202'  1A                           934   TP2         FCB     26
       =0010                        935               RADIX   16
0203'  2F 2F 2F 2F 2F 2F            936               FCB     2F,2F,2F,2F,2F,2F,2F,2F,2F,2F,2F,2F,2F
0209'  2F 2F 2F 2F 2F 2F
020F'  2F
0210'  2F 2F 2F 2F 2F 2F            937               FCB     2F,2F,2F,2F,2F,2F,2F,2F,2F,2F,2F,2F,2F
0216'  2F 2F 2F 2F 2F 2F
021C'  2F
       =000A                        938               RADIX   10
                                    939   *
                                    940   ********************************************************************************
```

```
                                942  ********************************************************************************
                                943  *
                                944  *       SERVICE ROUTINES FOR DUMPS AND OTHER TEST PRINTS
                                945  *
                                946  *       NOTE:  the following routines are strung together to save space
                                947  *              at the expense of some readability
                                948  *
                                949  *       output (D) as 4 hex digits
                                950  *
021D'  34 04                    951  OUT4    PSHS    B               save lo-order
021F'  BD 0224'                 952          JSR     OUT2            output hi-order
0222'  35 02                    953          PULS    A               get lo-order back
                                954  *       drop into out2 to output lo-order byte
                                955  *
                                956  *       output (A) as 2 hex digits
                                957  *
0224'  34 02                    958  OUT2    PSHS    A               save
0226'  44                       959          LSRA                    right justify hi-digit
0227'  44                       960          LSRA
0228'  44                       961          LSRA
0229'  44                       962          LSRA
022A'  BD 022F'                 963          JSR     HEX             output A bits 7-4
022D'  35 02                    964          PULS    A
                                965  *       drop into HEX to output lo-digit
                                966  *
                                967  *       output a (hex) digit from A3-A0
                                968  *
022F'  84 0F                    969  HEX     ANDA    #$F             mask
0231'  8B 20                    970          ADDA    #DBC_0          convert to DBC
0233'  81 29                    971          CMPA    #DBC_9          digit 0 to 9?
0235'  2F 02                    972          BLE     10$             skip if yes
0237'  8B 76                    973          ADDA    #$A0-$2A        convert to upper DBC for A through F
0239'                           974  10$
                                975  *       drop into CHAR to output the digit
                                976  *
                                977  *       output DBC char in A to the buffer
                                978  *
0239'  BD 000C*                 979  TSTBYT  JSR     DEVOUT          no, place char in buffer
023C'  39                       980          RTS
                                981  *
                                982  * 'TSTPUT' output test print line to device
                                983  *
                                984  * Entry : Y point to 'TSTBUF'
                                985  *
                                986  * Exit  : Y points to 'TSTBUF'
                                987  *
023D'  E6 A4                    988  TSTPUT  LDB     ,Y              Load byte count
023F'  27 0E                    989          BEQ     255$            No line to output
                                990  *
0241'  5C                       991          INCB                    Increment to next print position
0242'  E7 3D                    992          STB     -3,Y            Save end print position
0244'  C6 01                    993          LDB     #1              Load first print position
```

```
0246' E7 3C                994           STB     -4,Y            Save begin print position
0248' 6F 3E                995           CLR     -2,Y            Use device default print density
                           996+          CALL    DEVPUT          Output test print buffer to device
024A' BD 000D*            1005A          JSR     DEVPUT          (GO TO DEVPUT)
024D' 6F A4               1025           CLR     ,Y              Null test print buffer
024F' 39                  1026   255$    RTS                     Return
                          1027   *
                          1028   *       set up the test buffer for use
                          1029   *
0250' 108E 0013'          1030   SETBUF  LDY     #TSTBUF         buffer address is "always" in Y
0254' C6 F0               1031           LDB     #-16            number of bytes to clear -1
0256' 6F A5               1032   1$      CLR     B,Y             clear a byte
0258' 5C                  1033           INCB
0259' 2F FB               1034           BLE     1$              loop if not yet done
025B' 39                  1035           RTS                     return w/control clear and bufadr in Y
                          1036   *
                          1037   *       generate a "new_line"  --  CR-LF
                          1038   *
025C' 34 30               1039   NL      PSHS    Y,X             save X,Y
                          1040+          CALL    DEVCR           CR
025E' BD 000E*            1049A          JSR     DEVCR           (GO TO DEVCR)
                          1069+          CALL    DEVLF           LF
                          1074A          EXTERN  DEVLF
0261' BD 000F*            1078A          JSR     DEVLF           (GO TO DEVLF)
0264' 35 30               1098           PULS    X,Y             restore X,Y
0266' 39                  1099           RTS
                          1100   *
                          1101   *       routine checks for TEST switch on front panel pressed
                          1102   *               presense causes a return of FALSE. in A with Z set
                          1103   *               absense causes a return of TRUE. in A with Z clear
                          1104   *               SWFLAG is always cleared by this routine
                          1105   *
0267' B6 0008*            1106   CONCHK  LDA     SWFLAG          check for front parel switch
026A' 43                  1107           COMA
026B' 26 08               1108           BNE     3$
                          1109   *
                          1110   *       lda     #false.         switch flag set, prepare to clear it
                          1111   *
026D' F6 0000"            1112           LDB     SWVEC+3         what switch was it?
0270' B7 0008*            1113           STA     SWFLAG          clear the flag
0273' C1 07               1114           CMPB    #7              is switch code = TEST?
0275' 26 F0               1115           BNE     CONCHK          if switch was not TEST, go try again
                          1116   *
0277' 4D                  1117           TSTA                    set condition code (A remains FALSE.)
0278' 39                  1118   3$      RTS
                          1119   *
                          1120   ********************************************************************************
                          1121   *
                          1122           END
```

ADCOD	0008		AUTDMP	000A'IN		BUSY	0001 EX		CALL	Macro		CONCHK	0267'IN
DBC$C	00A2		DBC$K	00AA		DBC$SP	0010		DBC$X	00B7		DBC$0	0020
DBC$9	0029		DEVCR	000E EX		DEVLF	000F EX		DEVMCN	0002 EX		DEVOUT	000C EX
DEVPUT	0000 EX		DEVTST	0008 EX		DMPEND	00F2'		DUMP	00F6'		DVTST	0032'
ENAFLG	0003 EX		EXSTAT	0004 EX		FALSE.	0000		FDUMP	00BF'		GDUMP	0000'
HEX	022F'		LSTADR	0001'		LSTCOM	0005 EX		LSTORD	0006 EX		MEMTST	0036'
NL	025C'		NULL	0031'		OUT2	0224'		OUT4	021D'		PIDENT	0007 EX
PNTLIN	0144'		SDUMP	00E6'		SETBUF	0250'		STAT	0088'		STAT1	008E'
SWFLAG	0008 EX		SWVEC	0009 EX		TEST	0000'IN		TPRNT	0171'		TP1	01A1'
TP2	0202'		TRUE.	FFFF		TSTBUF	0013'IN		TSTBYT	0239'		TSTPUT	023D'
TSTTAB	001D'		TSWIT	0015'		VERS	0037'		VERSIO	000A EX		.RAMB.	2000
.RAME.	27FF		.ROMB.	C000		.ROME.	FFFF						

No errors detected

Cross reference listing (MREF version 4.5)

Symbol Refs (# = definition $ = write <blank> = read)

```
.RAMB.     69#    137     552
.RAME.     70#    137     553
.ROMB.     71#    136     423
.ROME.     72#    136     426
1$       1032#   1034
10$       263     265#    572#    599     603     740#    803
                  817#    909     972     974#
11$       267#    298
20$       260     301#    587#    590
255$      580     611#    989    1026#
3$       1108    1118#
30$       425#    427     588     594#
40$       575     607#
99$       172     174#
ADCOD     142#    171
AUTOMP     61     170#
BUSY       49#    508
CALL      212     268     304     334     364     394     431
                  477     624     653     682     711     743
                  772     818     847     877     996    1040
                 1069
CONCHK     62     574     908    1106#   1115
DBC_0     144#    970
DBC_9     145#    971
DBC_C     146#    333
DBC_K     147#    363
DBC_SP    143#    303     393     430     623     742
DBC_X     148#    476
DEVCR     852#    856    1049
DEVLF    1074#   1078
DEVMDN     50#    301
DEVOUT    273#    277     313     343     373     403     440
                  486     633     662     691     720     752
                  781     979
DEVPUT    823#    827     886    1005
DEVTST    217#    221
DMPEND    535     547     555#
DUMP      533     545     554     564#
DVTST     193     212#
ENAFLG     51#    510
```

Cross reference listing (MREF version 4.5)

Symbol Refs (# = definition $ = write <blank> = read)

```
EXSTAT     52#    506
FALSE.    141#    616     616     616     616     616     616
                  617
FDUMP      173     191     528#
GDUMP      156#    565$    607$    611
HEX        302     509     511     514     517     963     969#
LSTADR     157#    566$    579     587
LSTCOM      53#    515
LSTORD      54#    512
MEMTST     194     247#
NL         254     474     519     520     555     567     602
                   806     816     906     907    1039#
NULL       195     196     206#
OUT2       507     741     952     958#
OUT4       429     622     951#
PIDENT      55#    530     539     543
PNTLIN     572     621#
SDUMP      192     540     551#
SETBUF     253     473    1030#
STAT       190     473#
STAT1      460     476#
SWFLAG      56#   1106    1113$
SWVEC       57#    179    1112
TEST        63     164#
TP1        817     919#
TP2        876     934#
TPRNT      197     816#
TRUE.     140#     564     616     616     617
TSTBUF      64     159#    571    1030
TSTBYT     979#
TSTPUT     518     805     988#
TSTTAB     166     188#
TSWIT      164     170     179#
VERS       188     189     253#    528     551
VERSIO      58#    258
```