MVP 2200 Bootstrap Prom Listings

```
                * PROM set for Wang 2200 MVP systems
                *
                *       12/21/82
                *
                *       Computer Concepts Corporation
                *       8001 West 63rd Street
                *       Shawnee Mission ,KS  66202
                *
                *       Compiled by Paul Szudzik

0001            HARD:RESET      EQU     $0001
0003            START:ADDRESS   EQU     $0003
000E            SF'15RESETVECT  EQU     $000E

                *       The first 16 locations of the Prom are considered
                *       inviolate.  Lower memory may use these locations
                *       to vector to routines in Prom.
                *
                *       When first powered on, then system executes code starting at
                *       location 8003.  The first code jumps to POWERON:VECT and
                *       begins execution of the diagnostic.  For ease of entry, the
                *       POWERON vector is around $8131.
                *

                        ORG     $8000

8000 DE2E80     ERR:PARITY      JMP     PE:IN:CM        * PARITY ERROR CONTROL MEMORY
8001 5C0100     RESETV          JMP     HARD:RESET
8002 DE2780     ERR:DATA:DM     JMP     PE:IN:DM        * VECTOR FOR PARITY ERROR DATA MEM
8003 5D3180     POWERON         JMP     POWERON:VECT    * VECTOR FOR POWER ON

                *       Newer Proms from Wang contain an imbedded serial
                *       number at these locations.  These locations are
                *       NOT included in the checksum so may be changed
                *       at will without having to play with them

8004 DC0480     SERIAL#         JMP     SERIAL#
8005 5C0580                     JMP     .               * 2nd byte of serial

                *       Null locations

8006 5C0680                     JMP     .
8007 DC0780                     JMP     .
8008 DC0880                     JMP     .
8009 5C0980                     JMP     .
800A 5C0A80                     JMP     .

800B 5E4C80                     JMP     KEY:SF
                *
                *       The next instruction is purposely set to wrong parity
                *       to test the Wang CPU.
                *
                        WPAR

800C DC0C80     PARITY:TEST     JMP     .

                        EPAR
```

```
800D DC8A80   VSIZE:DM        JMP     SIZE:DM
800E DD2880   ERR:CHECKDM     JMP     DISP:VEDM
800F 5D2F80   ERR:CHECKDM     JMP     DISP:VEDM

              *       Sends one space to the selected IO device

8010 208E0F   OUTPUTSPACE     SET     K < $20         * OUTPUT A SPACE CODE

              *       Sends the character in K to the selected IO device

8011 68118D   OUTPUTK         BFL     8,SH    OUTPUTK * WAIT FOR READY
8012 548380   FORCEOUT[K]     JSR     PDELAY2         * DELAY FIRST
8013 978200                   CIO     08S             * SEND CHARACTER
8014 DC8380                   JMP     PDELAY2         * WAIT AGAIN

              *       Sends PH PL as 4 hex digits to selected IO

8015 200F0F   OUT4            SET     DUM < $00       * DISPLAYS PH - PL
8016 200009   DISPLAY:PHPL    IOR     R0 < 0,PH       * FIRST PH
8017 541980                   JSR     OUT2R0

              *       If entered here, sends PL as two hex digits to IO

8018 A00008   OUT2PL          IOR     R0 < 0,PL       * SEND PL OUT
8019 A0C19F   OUT2R0          SET     R1 < $39
801A 0C4201                   SDC  AH8H  R2 < R0,R1
8018 542080                   JSR     HEXADJUST
801C 884201                   SDC  ALBH  R2 < R0,R1
801D DC2080                   JMP     HEXADJUST

801E B00000   NULL:CR         OR      R0 < R0,R0
801F 5C2480                   JMP     CRLF

              *       Adjusts character in R2 to a hex A-F if required

8020 C82221   HEXADJUST       BLER    R2,R1   NOT:A:HEX
8021 AC0272                   IADD    R2 < 7,R2       * ADJUST TO HEX A-F
8022 200E02   NOT:A:HEX       IOR     K < 0,R2
8023 5C1180                   JMP     OUTPUTK

              *       Sends Carriage Return and Line Feed

8024 200EDF   CRLF            SET     K < $0D
8025 D41180                   JSR     OUTPUTK
8026 A00EAF                   SET     K < $0A
8027 5C1180                   JMP     OUTPUTK

              *       The character in PH is sent to the IO device the
              *       number of times contained in PL

8028 A00E09   SEND:PH         IOR     K < 0,PH
8029 D41180                   JSR     OUTPUTK
802A D828F8   REPEATOUTK      BNR     00-,PL  SEND:PH
802B 87800F                   RTS

              *       Sends three bytes to IO. K, PH and PL

802C D41180   OUTKPHPL        JSR     OUTPUTK         * SEND K OUT
802D A00E09   OUTPHPL         IOR     K < 0,PH        * SEND PH
```

```
802E D41180                    JSR     OUTPUTK
802F 200E08                    IOR     K ( 0,PL        * SEND PL
8030 5C1180                    JMP     OUTPUTK

8031 21430F  DISPLAY:ERROR     SET     R3 ( $50        * PExM
8032 21025F                    SET     R2 ( $45
8033 2100DF                    SET     R0 ( $40
8034 05800F                    TPS
8035 004713                    SDC  AL8L  R7 ( R1,R3
8036 783923                    BNL     2,R3     K8039
8037 7C3953                    BNH     5,R3     K8039
8038 A00787                    IOR     R7 ( 8,R7
8039 54A780  K8039             JSR     DELAY:2000
803A 548080                    JSR     SELECTCRT
803B 54A780                    JSR     DELAY:2000
803C 190050                    LPI     80              * CLEAN LINE OUT
803D 542A80                    JSR     REPEATOUTK
803E 54A780                    JSR     DELAY:2000
803F A00E5F                    SET     K ( 5
8040 190107                    LPI     $107
8041 542C80                    JSR     OUTKPHPL        * HOME AND RING BELL
8042 998050                    LPI     $2050
8043 542A80                    JSR     REPEATOUTK      * 80 SPACES
8044 200E1F                    SET     K ( $01
8045 198A2A                    LPI     $2A2A           * HOME AND **
8046 542C80                    JSR     OUTKPHPL
8047 D41180                    JSR     OUTPUTK         * ANOTHER *
8048 541080                    JSR     OUTPUTSPACE
8049 A14E3F                    SET     K ( $53
804A 1B4953                    LPI     $5953           * 'SYS'
804B 542C80                    JSR     OUTKPHPL
804C 214E4F                    SET     K ( $54
804D 9B054D                    LPI     $454D           * 'TEM'
804E 542C80                    JSR     OUTKPHPL
804F 541080                    JSR     OUTPUTSPACE
8050 180552                    LPI     $4552
8051 D42D80                    JSR     OUTPHPL         * 'ER'
8052 180F52                    LPI     $4F52
8053 542C80                    JSR     OUTKPHPL        * 'ROR'
8054 541080                    JSR     OUTPUTSPACE
8055 A08E8F                    SET     K ( $28
8056 8208E2                    ORX     PL ( 00+,R2     * '(' AND R2,R3
8057 542C80                    JSR     OUTKPHPL
8058 0208E0                    ORX     PL ( 00+,R0     * R0,R1
8059 D42D80                    JSR     OUTPHPL
805A 541080                    JSR     OUTPUTSPACE
805B E85D47                    BFL     4,R7     K805D
805C 57EC80                    JSR     DISP:BANK
805D 80800F  K805D             TSP
805E 541680                    JSR     DISPLAY:PHPL
805F FC6907                    BNH     0,R7     K8069
8060 787137                    BNL     3,R7     RIGHT:PAREN
8061 541080                    JSR     OUTPUTSPACE
8062 DC6480                    JMP     DISPLAY:CM
8063 DE2E80                    JMP     PE:IN:CM
8064 D4AF80  DISPLAY:CM        JSR     READ:CM
8065 A0000E                    IOR     R0 ( 0,K
8066 541980                    JSR     OUT2R0
8067 541680                    JSR     DISPLAY:PHPL
8068 5C7180                    JMP     RIGHT:PAREN
```

```
8069 7C7127   K8069            BNH     2,R7      RIGHT:PAREN
806A 541080                    JSR     OUTPUTSPACE
806B 706F47                    BEL     4,R7      K806F
806C 200006                    IOR     R0 < 0,R6
806D 541980                    JSR     OUT2R0
806E F071C7                    BEL     $C,R7     RIGHT:PAREN
806F 8208E4   K806F             ORX     PL < 00+,R4
8070 541680                    JSR     DISPLAY:PHPL
8071 208E9F   RIGHT:PAREN      SET     K < $29         *
8072 D41180                    JSR     OUTPUTK
8073 541080                    JSR     OUTPUTSPACE
8074 208EAF                    SET     K < $2A         * '*'
8075 198A2A                    LPI     $2A2A           * '**'
8076 542C80                    JSR     OUTKPHPL
8077 5C2480                    JMP     CRLF

8078 200F0F                    SET     DUM < $00
8079 DE2780                    JMP     PE:IN:DM

807A A0001F   DR01WITHR0       SET     R0 < 1
8078 87800F                    RTS

807C 8381AF   SELECTAR1AL      XPA     AR 1A
807D 200E08                    IOR     K < 0,PL
807E 8381AF                    XPA     AR 1A
807F 5C8180                    JMP     SELECTK8

8080 A00E5F   SELECTCRT        SET     K < 5           * ADDRESS OF CRT
8081 178C00   SELECTK8        CIO     CA8,ABS         * SELECT DEVICE IN K
8082 548380   PDELAY1          JSR     PDELAY2
8083 D48480   PDELAY2          JSR     PDELAY3
8084 548580   PDELAY3          JSR     PDELAY4
8085 200F0F   PDELAY4          SET     DUM < 0
8086 200F0F                    SET     DUM < 0
8087 87800F                    RTS

              *       Sizes either Control Memory or Data Memory depending
              *       on entry.  If Data memory is being sized, only the
              *       Bank currently selected is tested.  On return, the
              *       highest address + 1 is in R1,R0 and AR 00.

8088 190FFF   SIZE:DM           LPI     $0FFF
8089 5CBD80                    JMP     BEGIN:SIZER
808A 994FFF   SIZE:DM          LPI     $1FFF
8088 A20D0D                    IOR     SH < $80,SH     * SET NO DM ERRORS
808C 2ACDF0                    IAND    SH < $BF,SH     * CLEAR DM ERROR
808D 860000   BEGIN:SIZER      XORX    R0 < R0,R0      * CLEARS R0,R1
808E A00209                    IOR     R2 < 0,PH
808F 81800F   NEXT:LOC         TPA     AR 00
8090 7C9612                    BNH     1,R2      SIZE:CM0 * Control Memory
8091 A16FAF                    SET     DUM < $5A ,W1
8092 A01F0F                    SET     DUM < 0   ,RD
8093 A02F0F                    SET     DUM < $00 ,W1
8094 200E0B                    IOR     K < 0,CH        * K = DM LOCATION
8095 5C9A80                    JMP     CHECK:RESULT
8096 A14EAF   SIZE:CM0         SET     K < $5A
8097 D4B180                    JSR     WRITE:CM        * WRITE CM
8098 88800F                    TAP     AR 00
8099 D4AF80                    JSR     READ:CM         * READ BACK
809A D0A3EE   CHECK:RESULT     BER     00+,K     SIZE:FOUND
```

```
809B 7C9DFE                     BNH     $F,K      MEM:PRESENT
809C F0A3FE                     BEL     $F,K      SIZE:FOUND
809D 8B800F   MEM:PRESENT       TAP     AR 00
809E 0200E8                     ORX     RB < 00+,PL
809F 18C929                     ADC     PH < R2,PH  ,CS
80A0 74A212                     BEH     1,R2      CHECK:DONE
80A1 74A389                     BEH     8,PH      SIZE:FOUND
80A2 588F29   CHECK:DONE        BNR     R2,PH     NEXT:LOC * Test next location
80A3 7CA612   SIZE:FOUND        BNH     1,R2      CLEAN:EXIT
80A4 195FFF                     LPI     $1FFF ,RD
80A5 A8CDFD                     IAND    SH < $3F,SH     * MASK ERROR,  ENABLE DM
80A6 87800F   CLEAN:EXIT        RTS

              *       The next routine performs a time delay.  The contents of
              *       the PHPL pair are decremented till 0 is reached.
              *
80A7 998000   DELAY:2000        LPI     $2000
80A8 D48280   DELAY:[PHPL]      JSR     PDELAY1
80A9 58A8F8                     BNR     00-,PL    DELAY:[PHPL]
80AA FCA809                     BNH     0,PH      DELAY:[PHPL]
80AB 87800F                     RTS


80AC A00040   OR04WITHR0        IOR     R0 < 4,R0
80AD 87800F                     RTS


              *       Control Memory may be read by using this subroutine.
              *       On entry, PHPL is the location to read data.
              *       On return, K = MSD, PH = MMD and PL = LSD.

80AE 200F0F   RCM:DELAYED       SET     DUM < 0
80AF 05B00F   READ:CM           TPS
80B0 878600                     RTS     ,RC

              *       Control Memory may be written to by this routine.
              *       On entry, the following conditions must have been set:
              *
              *       1: AR 00 contains the MMD and LSD digits to write
              *       2: K contains then MSD
              *       3: PHPL points to the location to write
              *
              *       On return, K register will have been ones' complemented.

80B1 A7CEFE   WRITE:CM          IXOR    K < $FF,K       * COMPLEMENT K
80B2 05B00F                     TPS
80B3 8B800F                     TAP     AR 00
80B4 078400                     RTS     ,WC

              *       The following routine tests one location in CM
              *       The entry points are as follows:
              *
              *       1: PHPL contains address to test
              *       2: K register contains the MSD
              *       3: AR 00 contains the MMD and LSD data
              *
              *       Failure will result in Error display but no return.

80B5 05B00F   TEST:ONE:CM       TPS
80B6 A0000E                     IOR     R0 < 0,K
```

```
80B7 D48180                     JSR     WRITE:CM
80B8 8D800F                     TSP
80B9 05800F                     TPS
80BA D4AF80                     JSR     READ:CM
80BB 040E0E                     XOR     K < R0,K
80BC 0200E8                     ORX     R0 < 00+,PL
80BD 8B800F                     TAP   AR 00
80BE 060008                     XORX    R0 < R0,PL
80BF 8D800F                     TSP
80C0 D8C31F                     BNR     R1,00     CM:BITERROR
80C1 58C30F                     BNR     R0,00     CM:BITERROR
80C2 50A60E                     BER     R0,K      CLEAN:EXIT
80C3 A0060E   CM:BITERROR       IOR     R6 < 0,K
80C4 0204E0                     ORX     R4 < 00+,R0
80C5 21013F                     SET     R1 < $43        * 'C'
80C6 21032F   SETBITERROR       SET     R3 < $42        * 'B'
80C7 DD2D80                     JMP     SHOW:ERROR

              *       The location pointed to by PHPL pair is read. The contents
              *       of the location is then written to address 0001 of Control
              *       Memory, which is the Hard Reset vector.

80C8 D4AF80   REWRITERESETV     JSR     READ:CM
80C9 81800F                     TPA     AR 00
80CA 990001                     LPI     $0001
80CB 5CB180                     JMP     WRITE:CM

              *       The location pointed to by PHPL pair is read. The contents
              *       of the location is then written to address 000D of Control
              *       Memory, which is the SF'15 vector address.
              *
80CC D4AF80   REWRITE'15VEC     JSR     READ:CM
80CD 81800F                     TPA     AR 00
80CE 99000D                     LPI     $000D
80CF 5CB180                     JMP     WRITE:CM

              *       Clears 64K of memory in selected bank.  Locations 0,1 are
              *       set to FF00.
              *
80D0 8A00EE   CLEAR:DM          ANDX    R0 < 00+,K
80D1 990001                     LPI     $0001
80D2 23FFFF                     SET     DUM < $FF ,W2
80D3 802FEF                     OR      DUM < 00+,00 ,W1
80D4 C4D3D8                     BLRX    R0,PL     .-1
80D5 87800F                     RTS

              *       Test Control Memory. Pattern for test is 8000
              *       unless entered by TEST:CM[PHPL]


80D6 190000   TEST:CM            LPI     $0000
80D7 01801F   TEST:CM:[PHPL]    TPA     AR 01
80D8 57FD80                     JSR     VSIZE:CM        * SIZE CONTROL MEMORY 1ST
80D9 08801F                     TAP     AR 01
80DA A60209                     IXOR    R2 < $80,PH
80DB 81800F                     TPA     AR 00
80DC 020BE0                     ORX     PL < 00+,R0
80DD 200E02   TEST:NEXT:CM      IOR     K < 0,R2
80DE 548580                     JSR     TEST:ONE:CM     * Test location in CM
80DF 58DDF9                     BNR     00-,PH    TEST:NEXT:CM
```

```
80E0 FCDD08                     BNH     0,PL      TEST:NEXT:DM
80E1 87800F                     RTS

                *
                *       The next routine is interesting because it simply
                *       scans Data Memory looking for Parity
                *       errors.

80E2 540D80     SCAN:DM         JSR     VSIZE:DM        * SIZE DATA MEMORY
80E3 1FCFFF                     LPI     $FFFF
80E4 A2000D                     IOR     SH < $80,SH     * MASK DM ERRORS
80E5 000FEF     TEST:NEXT:DM    OR      DUM < 00+,00
80E6 801FEF                     OR      DUM < 00+,00 ,RD
80E7 E4024D                     BTH     4,SH      ERR:DATA:DM
80E8 58E519                     BNR     R1,PH     TEST:NEXT:DM
80E9 58E508                     BNR     R0,PL     TEST:NEXT:DM
80EA A8CDFD                     IAND    SH < $3F,SH
80EB 20000F     CLEAR:R0:EXIT   SET     R0 < 0
80EC 87800F                     RTS

                *
                *       The next routine handles the checksums in Control Memory
                *       Location 000F contains the highest address to checksum.
                *       If entered through PROM:CHECKS, the following must
                *       be set:
                *
                *       1: AR 03 is ending address
                *       2: AR 02 is modulo location of checksums
                *       3: AR 00 is starting address

80ED 19000F     CHECKSUMS       LPI     $F              * Highest location to checksum is here
80EE D4AF80                     JSR     READ:CM
80EF 50EBE9                     BER     00+,PH    CLEAR:R0:EXIT * =0 = NO CHECKS
80F0 81803F                     TPA     AR 03           * Save top location here
80F1 190000                     LPI     $0000
80F2 81800F                     TPA     AR 00           * Starting address
80F3 28CDED                     IAND    SH < $FE,SH
80F4 2009FF                     SET     PH < $F
80F5 01802F     K80F5           TPA     AR 02           * Modulo locations of Checksums
80F6 060222     PROM:CHECKS     XORX    R2 < R2,R2
80F7 060444                     XORX    R4 < R4,R4
80F8 860666                     XORX    R6 < R6,R6
80F9 B8800F                     TAP     AR 00
80FA 01820F     K80FA           TPA     AR 00 ,+1
80FB D4AF80                     JSR     READ:CM
80FC D74D80                     JSR     CALC:PARITY
80FD D1040F                     BER     R0,00     NOPARITY:ERR
                *
                *       A parity error occurred. However it may be the Test address.
                *       Check to see if Address 800C was defective
                *
80FE 03800F                     XPA     AR 00
80FF FE2F89                     BNH     8,PH      PE:IN:CM+1
8100 FA2F09                     BNL     0,PH      PE:IN:CM+1
8101 FE2F08                     BNH     0,PL      PE:IN:CM+1
8102 FA2FD8                     BNL     $0,PL     PE:IN:CM+1
8103 03800F                     XPA     AR 00
8104 84077E     NOPARITY:ERR    XOR     R7 < R7,K
8105 060558                     XORX    R5 < R5,PL
8106 1A8222                     ADCX    R2 < R2,R2 ,CC
```

```
8107 180444                 ADC     R4 ( R4,R4
8108 380202                 IADC    R2 ( 0,R2
8109 9A022B                 ADCX    R2 ( R2,PL
810A 18044E                 ADC     R4 ( R4,K
810B 88800F                 TAP     AR 00
810C F8FAE8                 BNL     $E,PL     K80FA
810D FCFAF8                 BNH     $F,PL     K80FA
810E 08802F                 TAP     AR 02           * Check for modulo now
810F 200009                 IOR     R0 ( 0,PH
8110 88800F                 TAP     AR 00
8111 58FA09                 BNR     R0,PH     K80FA
8112 01820F                 TPA     AR 00 ,+1       * Yup, Check Checksums now
8113 D4AF80                 JSR     READ:CM
8114 59294E                 BNR     R4,K      K8129
8115 592939                 BNR     R3,PH     K8129
8116 592928                 BNR     R2,PL     K8129
8117 84077E                 XOR     R7 ( R7,K
8118 060558                 XORX    R5 ( R5,PL
8119 88800F                 TAP     AR 00
811A 01820F                 TPA     AR 00 ,+1
811B D4AF80                 JSR     READ:CM
811C A7C7F7                 IXOR    R7 ( $FF,R7
811D A7C6F6                 IXOR    R6 ( $FF,R6
811E A7C5F5                 IXOR    R5 ( $FF,R5
811F 59297E                 BNR     R7,K      K8129
8120 592969                 BNR     R6,PH     K8129
8121 D92958                 BNR     R5,PL     K8129
8122 88800F                 TAP     AR 00
8123 200009                 IOR     R0 ( 0,PH
8124 88803F                 TAP     AR 03           * See if finished
8125 D0E809                 BER     R0,PH     CLEAR:R0:EXIT
8126 08802F                 TAP     AR 02
8127 AC4909                 IADD    PH ( $10,PH
8128 5CF580                 JMP     K80F5
8129 88800F   K8129           TAP     AR 00           * ERROR
812A 800FFF                 OR      DUM ( 00-,00
812B 21013F   DISP:VECM        SET     R1 ( $43        * 'C'
812C 21436F   DISP:VExM        SET     R3 ( $56        * 'V'
812D 543280   SHOW:ERROR      JSR     DISPLAY:ERROR+1
812E 5E3780                 JMP     K8237

812F A1014F   DISP:VECM        SET     R1 ( $44        * 'D'
8130 5D2C80                 JMP     DISP:VExM

              *       The main purpose of the Proms is to test the integrity
              *       of the 2200 CPU and memory.  The next routine checks all
              *       combinations of registers and instructions to verify that
              *       the 2200 can load and execute programs.
              *

8131 A0002D   POWERON:VECT    IOR     SH ( 2,SH       * SET BUSY FOR DEVICES
8132 190000                 LPI     0
8133 190000                 LPI     0
8134 190000                 LPI     0
8135 A00E5F                 SET     K ( 5           * SELECT CRT
8136 17BC00                 CIO     CA8,ABS
8137 190000                 LPI     0
8138 190000                 LPI     0
8139 E9398D                 BFL     8,SH      .     * WAIT TILL READY
813A A00E3F                 SET     K ( 3           * CLEAR SCREEN
```

```
813B 978200                 CIO     OBS
813C E93C8D                 BFL     8,SH      .     * WAIT TILL READY
813D A10EDF                 SET     K < $4D         * 'M'
813E 978200                 CIO     OBS
813F E93F8D                 BFL     8,SH      .
8140 5C0CB0                 JMP     PARITY:TEST
8141 210EFF   PASSED:PARITY SET     K < $4F         * 'O'
8142 978200                 CIO     OBS
8143 69438D                 BFL     8,SH      .     * WAIT FOR READY

              *       Check subroutine and basic ALU instructions

8144 547A80                 JSR     OR01WITHR0      * BASIC REGISTER TEST
8145 D4AC80                 JSR     OR04WITHR0
8146 D772B0                 JSR     OR20WITHR0
8147 D77480                 JSR     OR80WITHR0      * ANSWER = A5
8148 F048A0                 BNH     $A,R0     .     * ERROR HANGS
8149 F94950                 BNL     $5,R0     .
814A 184505                 LPI     $5505           * 'U' & CURSOR ON
814B D42D80                 JSR     OUTPHPL
814C 9D0003                 LPI     $8003
814D 57E380                 JSR     WRITE1ST16DM    * WRITE LOCATIONS
814E 992000                 LPI     0 ,W1           * ZERO DM LOC 0
814F 193000                 LPI     0 ,W2           * ZERO DM LOC 1
8150 991000                 LPI     0 ,RD           * READ BACK
8151 A8CDFD                 IAND    SH < $3F,SH     * MASK ERRORS
8152 A10EEF                 SET     K < $4E         * 'N'
8153 D41180                 JSR     OUTPUTK

              *       Test all general registes with ripple patterns

8154 1D[illegible]          LPI     $5AA5
8155 0200E8   K8155         ORX     R0 < 00+,PL
8156 0202E0                 ORX     R2 < 00+,R0
8157 8204E2                 ORX     R4 < 00+,R2
8158 0206E4                 ORX     R6 < 00+,R4
8159 5A3A79                 BNR     R7,PH     ERROR
815A 5A3A68                 BNR     R6,PL     ERROR
815B 0208E5                 ORX     PL < 00+,R5     * SHIFT PATTERN
815C FD5559                 BNH     5,PH      K8155
815D 214E4F                 SET     K < $54         * 'T'
815E D41180                 JSR     OUTPUTK

              *       Test auto increment of pHPL pair

815F 20000F                 SET     R0 < 0
8160 190000                 LPI     0
8161 000FEF                 OR      DUM < 00+,00
8162 DA3A09                 BNR     R0,PH     ERROR
8163 7E3A08                 BNH     0,PL      ERROR
8164 7A3A18                 BNL     1,PL      ERROR
              *       Test auto decrement feature
8165 0C80FF                 SBC     R0 < 00-,00 ,CC
8166 800FFF                 OR      DUM < 00-,00
8167 DA3A09                 BNR     R0,PH     ERROR
8168 5A3A08                 BNR     R0,PL     ERROR
8169 541080                 JSR     OUTPUTSPACE

              *       Test increment and decrement of Auxillary Registers
```

```
816A 190000                     LPI     0
816B 81800F                     TPA     AR 00
816C 81821F                     TPA     AR 01  ,+1
816D 01C22F                     TPA     AR 02  ,-1
816E 08802F                     TAP     AR 02
816F 0200E8                     ORX     R0 < 00+,PL
8170 1FCFFF                     LPI     $FFFF
8171 5A3A19                     BNR     R1,PH     ERROR
8172 5A3A0B                     BNR     R0,PL     ERROR
8173 9AC0E0                     ADCX    R0 < 00+,R0 ,CS
8174 8B800F                     TAP     AR 00
8175 5A3A19                     BNR     R1,PH     ERROR
8176 5A3A08                     BNR     R0,PL     ERROR
8177 9AC0E0                     ADCX    R0 < 00+,R0 ,CS
8178 0B801F                     TAP     AR 01
8179 5A3A19                     BNR     R1,PH     ERROR
817A 5A3A08                     BNR     R0,PL     ERROR
817B A14E3F                     SET     K < $53         * 'S'
817C D41180                     JSR     OUTPUTK
                *       ADD instructions
817D 2D40AF                     IADD    R0 < $5A,00
817E AEB150                     IADD    R1 < $A5,R0
817F FE3A50                     BNH     5,R0      ERROR
8180 7A3AA0                     BNL     $A,R0     ERROR
8181 7E3AF1                     BNH     $F,R1     ERROR
8182 FA3AF1                     BNL     $F,R1     ERROR
8183 AD4281                     IADD    R2 < $5B,R1
8184 5A3A02                     BNR     R0,R2     ERROR
8185 98C321                     ADC     R3 < R2,R1 ,CS
8186 DA3A30                     BNR     R3,R0     ERROR
8187 9B8421                     ADC     R4 < R2,R1 ,CC
8188 18C340                     ADC     R3 < R4,R0 ,CS
8189 7E3A54                     BNH     5,R4      ERROR
818A FA3A94                     BNL     9,R4      ERROR
818B 7E3AB3                     BNH     $B,R3     ERROR
818C FA3A43                     BNL     4,R3      ERROR
818D A00D1D                     IOR     SH < $01,SH     * SET CARRY
818E 3BC3FF                     IADC    R3 < $FF,00
818F DA3A3F                     BNR     R3,00     ERROR
8190 28CDE0                     IAND    SH < $FE,SH     * CLEAR CARRY
8191 3BC3F3                     IADC    R3 < $FF,R3
8192 FE3AF3                     BNH     $F,R3     ERROR
8193 7A3AF3                     BNL     $F,R3     ERROR
                *       Subtract instructions
8194 0CC430                     SBC     R4 < R3,R0  ,CS
8195 7E3AA4                     BNH     $A,R4     ERROR
8196 FA3A54                     BNL     $5,R4     ERROR
8197 20071F                     SET     R7 < 1
8198 0C877F                     SBC     R7 < R7,00 ,CC
8199 5A3A7F                     BNR     R7,00     ERROR
819A 1900FF                     LPI     $FF
8198 8201EB                     ORX     R1 < 00+,PL
819C 1AC3E1                     ADCX    R3 < 00+,R1 ,CS
8190 990100                     LPI     $100
819E 5A3A49                     BNR     R4,PH     ERROR
819F 5A3A38                     BNR     R3,PL     ERROR
81A0 20000F                     SET     R0 < 0
81A1 9EB33F                     S8CX    R3 < R3,00 ,CC
81A2 DA3A42                     BNR     R4,R2     ERROR
81A3 5A3A31                     BNR     R3,R1     ERROR
```

```
81A4 A14E9F                    SET     K < $59         * 'Y'
81A5 D41180                    JSR     OUTPUTK

             *       Stack depth instructions
81A6 194111                    LPI     $1111
81A7 8201E8                    ORX     R1 < 00+,PL
81A8 05800F   K81A8            TPS
81A9 98C00F                    ADC     R0 < R0,00 ,CS
81AA 1A8818                    AOCX    PL < R1,PL ,CC
81AB 7DA860                    BNH     6,R0      K81A8
81AC 0203E8                    ORX     R3 < 00+,PL
81AD 80800F   K81AD            TSP
81AE BEC331                    SBCX    R3 < R3,R1  ,CS
81AF 5A3A49                    BNR     R4,PH     ERROR
81B0 5A3A38                    BNR     R3,PL     ERROR
81B1 0C800F                    SBC     R0 < R0,00 ,CC
81B2 59AD0F                    BNR     R0,00     K81AD
81B3 A14E3F                    SET     K < $53         * 'S'
81B4 D41180                    JSR     OUTPUTK
             *       Immediate instruction test
81B5 28CDED                    IAND    SH < $FE,SH     * CLEAR CARRY
81B6 B1400F                    IDAC    R0 < $50,00
81B7 B0011F                    IDAC    R1 < $01,00
81B8 354210                    IDSC    R2 < $51,R0
81B9 0A3A21                    BNR     R2,R1     ERROR
81BA A00D1D                    IOR     SH < $01,SH     * SET CARRY
81BB B40221                    IDSC    R2 < $02,R1
81BC 5A3A2F                    BNR     R2,00     ERROR
             *       Decimal Register instructions
81BD 22409F                    SET     R0 < $99
81BE 108302                    DAC     R3 < R0,R2 ,CC
81BF DA3A03                    BNR     R0,R3     ERROR
81C0 10C33F                    DAC     R3 < R3,00 ,CS
81C1 EA3A1D                    BFL     1,SH      ERROR
81C2 DA3A3F                    BNR     R3,00     ERROR
81C3 1482E0                    DSC     R2 < 00+,R0 ,CC
81C4 0A3A12                    BNR     R1,R2     ERROR
81C5 14C433                    DSC     R4 < R3,R3 ,CS
81C6 5A3A40                    BNR     R4,R0     ERROR
81C7 214E4F                    SET     K < $54         * 'T'
81C8 D41180                    JSR     OUTPUTK
             *       Multiply instructions
81C9 2140AF                    SET     R0 < $5A
81CA A0C1CF                    SET     R1 < $3C
81CB 8C8250                    IMUL  ALBH  R2 < $25,R0
81CC FE3A12                    BNH     1,R2      ERROR
81CD FA3A92                    BNL     9,R2      ERROR
81CE 3C0271                    IMUL  ALBL  R2 < 7,R1
81CF 7E3A52                    BNH     5,R2      ERROR
81D0 7A3A42                    BNL     4,R2      ERROR
81D1 9CC302                    MUL   AH8H   R3 < R0,R2
81D2 7E3A13                    BNH     1,R3      ERROR
81D3 7A3A93                    BNL     9,R3      ERROR
81D4 1CB313                    MUL   ALBH   R3 < R1,R3
81D5 FE3A03                    BNH     0,R3      ERROR
81D6 7A3AC3                    BNL     $C,R3     ERROR
81D7 1C0213                    MUL   AL8L   R2 < R1,R3
81D8 7E3A92                    BNH     9,R2      ERROR
81D9 FA3A02                    BNL     0,R2      ERROR
81DA 210E5F                    SET     K < $45         * 'E'
```

```
81DB D41180                     JSR     OUTPUTK

                *       Shift Decimal Character instructions
81DC 2140AF                     SET     R0 ( $5A
81DD A2B15F                     SET     R1 ( $A5
81DE 044211                     SDC  AHBL  R2 ( R1,R1
81DF 5A3A20                     BNR     R2,R0     ERROR
81E0 0C4300                     SDC  AHBH  R3 ( R0,R0
81E1 FE3A53                     BNH     5,R3      ERROR
81E2 7A3A53                     BNL     5,R3      ERROR
81E3 004410                     SDC  ALBL  R4 ( R1,R0
81E4 0A3A41                     BNR     R4,R1     ERROR
81E5 084304                     SDC  ALBH  R3 ( R0,R4
81E6 FE3AA3                     BNH     $A,R3     ERROR
81E7 7A3AA3                     BNL     $A,R3     ERROR
81E8 A10EDF                     SET     K ( $4D         * 'M'
81E9 D41180                     JSR     OUTPUTK
                *       Checksum the Proms now to ensure integrity
81EA 1D0400                     LPI     $8400
81EB 81803F                     TPA     AR 03
81EC 9D0006                     LPI     $8006
81ED 81800F                     TPA     AR 00           * PREPARE FOR CHECKSUMS
81EE A2093F                     SET     PH ( $83
81EF 01802F                     TPA     AR 02
81F0 D4F680                     JSR     PROM:CHECKS
81F1 541080                     JSR     OUTPUTSPACE
                *       Clear all of Data Memory, existant or not
81F2 200C0F                     SET     SL ( $00
81F3 54D080     K81F3           JSR     CLEAR:DM
81F4 AC8C0C                     IADD    SL ( $20,SL     * BOOST BANK
81F5 FDF30C                     BNH     $00,SL    K81F3
81F6 A14E0F                     SET     K ( $50         * 'P'
81F7 D41180                     JSR     OUTPUTK

                *       Rewrite Reset Vector to point to KEY:SF routine and check
81F8 1D000B                     LPI     $8008
81F9 54C880                     JSR     REWRITERESETV
81FA 27C0FE                     IXOR    R0 ( $FF,K
81FB 8201EB                     ORX     R1 ( 00+,PL
81FC 990001                     LPI     $0001
81FD D4AF80                     JSR     READ:DM
81FE 5A3A0E                     BNR     R0,K      ERROR
81FF 5A3A29                     BNR     R2,PH     ERROR
8200 0A3A1B                     BNR     R1,PL     ERROR
8201 01801F                     TPA     AR 01
8202 8208EE                     ORX     PL ( 00+,K
8203 01802F                     TPA     AR 02
8204 210ECF                     SET     K ( $4C         * 'L'
8205 9B0154                     LPI     $4154           * 'AT'
8206 542C80                     JSR     OUTKPHPL
8207 180552                     LPI     $4552           * 'TER'
8208 542C80                     JSR     OUTKPHPL
8209 D63B80                     JSR     PRESS:RESET

                *       Now while waiting for the user to press RESET, do memory
                *       tests.

820A 1B4A5A     TEST:DM:DM      LPI     $5A5A           * MEMORY PATTERN
8208 04D780                     JSR     TEST:DM:[PHPL]
820C 1D85A5                     LPI     $A5A5
```

```
820D D4D780                     JSR     TEST:CM:[PHPL]
820E 540DB0                     JSR     VSIZE:DM        * SIZE DATA MEMORY

                *       Does the memory test in the background loop

820F A20D0D                     IOR     SH < $80,SH     * MASK PARITY ERRORS
8210 1AC8E0                     ADCX    PL < 00+,R0  ,CS
8211 B1B00F                     TPA     AR 00
8212 20060F     K8212           SET     R6 < 0
8213 88800F     K8213           TAP     AR 00
8214 2C0017                     IADD    R0 < 1,R7
8215 2C0110                     IADD    R1 < 1,R0
8216 8082FF     K8216           OR      R2 < 00-,00 ,CC
8217 B00FFF                     OR      DUM < 00-,00
8218 DA1B6F                     BNR     R6,00     K821B
8219 202F01                     IDR     DUM < 0,R1  ,W1
821A 203F00                     IOR     DUM < 0,R0  ,W2
8218 D76A80     K8218           JSR     READ:DM:2BYTES
821C A00001                     IDR     R0 < 00,R1
821D AC0111     K821D           IADD    R1 < 1,R1
821E 521D71                     BER     R7,R1     K821D
821F 5A1628                     BNR     R2,PL     K8216
8220 DA1629                     BNR     R2,PH     K8216
8221 AC0616                     IADD    R6 < 1,R6
8222 FA1336                     BNL     3,R6      K8213
8223 AC0717                     IADD    R7 < 1,R7
8224 FA1207                     BNL     0,R7      K8212
8225 DE0A80                     JMP     TEST:CM:DM      * REPEAT TEST

8226 5E4C80                     JMP     KEY:SF

8227 A1014F     PE:IN:DM        SET     R1 < $44        * 'D'
8228 543180                     JSR     DISPLAY:ERROR
8229 A20D0D                     IOR     SH < $80,SH
822A 2ACDFD                     IAND    SH < $BF,SH
822B 991000                     LPI     0 ,RD
822C 29CDFD                     IAND    SH < $7F,SH
822D 5E3780                     JMP     K8237

                *       Vectored here if an instruction was executed with wrong
                *       Parity.  The address of the error is pushed onto the
                *       stack by hardware.  This routine checks to see if the
                *       special address 800C was the error, which was from
                *       the Power on test.  Else an error is displayed.

822E 8D800F     PE:IN:CM        TSP
822F 800FFF                     OR      DUM < 00-,00
8230 0200E8                     ORX     R0 < 00+,PL
8231 90000C                     LPI     $800C
8232 DA3419                     BNR     R1,PH     .+2
8233 D14108                     BER     R0,PL     PASSED:PARITY
8234 0208E0                     ORX     PL < 00+,R0
8235 21013F                     SET     R1 < $43        * 'C'
8236 543180                     JSR     DISPLAY:ERROR
8237 10000B     K8237           LPI     $8008
8238 54C880                     JSR     REWRITERESETV
8239 D63880                     JSR     PRESS:RESET
823A DE3A80     ERROR           JMP     .

                *
```

```
                *       This routine displays the message 'PRESS RESET' on the
                *       screen on the second line.
                *
8238 99010A     PRESS:RESET     LPI     $010A           * HOME AND LF
823C D42D80                     JSR     OUTPHPL
823D 998050                     LPI     $2050           * 80 SPACES
823E 542A80                     JSR     REPEATOUTK
823F 200E1F                     SET     K < $01
8240 190A50                     LPI     $0A50           * HOME , LF AND 'P'
8241 542C80                     JSR     OUTKPHPL
8242 214E2F                     SET     K < $52         * 'R'
8243 9B0553                     LPI     $4553           * 'ES'
8244 542C80                     JSR     OUTKPHPL
8245 198052                     LPI     $2052           * 'S R'
8246 542C80                     JSR     OUTKPHPL
8247 9B0553                     LPI     $4553           * 'ES'
8248 D42D80                     JSR     OUTPHPL
8249 180554                     LPI     $4554           * 'ET'
824A D42D80                     JSR     OUTPHPL
824B 5C2480                     JMP     CRLF

824C A00C2F     KEY:SF          SET     SL < $2
824D 220D2D                     IOR     SH < $82,SH     * SET BUSY
824E 2ACDFD                     IAND    SH < $8F,SH
824F 991000                     LPI     0 ,RD
8250 548080                     JSR     SELECTCRT
8251 A00E3F                     SET     K < 3           * CLEAR SCREEN
8252 190548                     LPI     $054B           * CURSOR 'K'
8253 542C80                     JSR     OUTKPHPL
8254 210E5F                     SET     K < $45         * 'E'
8255 9B4920                     LPI     $5920           * 'Y '
8256 542C80                     JSR     OUTKPHPL
8257 984346                     LPI     $5346           * 'SF'
8258 D42D80                     JSR     OUTPHPL
8259 19873F                     LPI     $273F           * '?/              JSR     OUTPHPL
825A D42D80                     JSR     OUTPHPL
825B 541080     K825B           JSR     OUTPUTSPACE
825C A0850F                     SET     R5 < $20
825D A0860F                     SET     R6 < $20
825E 190000                     LPI     0
825F 81810F                     TPA     AR 10
8260 AB8C7C                     IAND    SL < $E7,SL
8261 56FA80                     JSR     INPUTC
8262 20070E                     IOR     R7 < 0,K
8263 56FA80                     JSR     INPUTC
8264 A0060E                     IOR     R6 < 0,K
8265 56FA80                     JSR     INPUTC
8266 A0050E                     IOR     R5 < 0,K
8267 A04C0C                     IOR     SL < $10,SL
8268 56FA80                     JSR     INPUTC
8269 5C0380                     JMP     POWERON

826A AC4000     LOOP            IADD    R0 < $10,R0
8268 5A800F                     BNR     R0,00     K8280
826C 8880CF                     TAP     AR 0C
826D 200408                     IOR     R4 < 0,PL
826E 8880FF                     TAP     AR 0F
826F 537648                     BER     R4,PL     PERR:D82
8270 D78180                     JSR     PREAD:SECTOR
8271 994F00                     LPI     $1F00
```

```
8272 5E7F80                    JMP     K827F

8273 2RCCDC    LOAD:PROGRAM    IAND    SL < $FD,SL
8274 547C80                    JSR     SELECTAR1AL
8275 A20D0D                    IOR     SH < $80,SH
8276 9D0003                    LPI     $8003
8277 57E380                    JSR     WRITE1ST16CM
8278 190000                    LPI     0
8279 8180FF                    TPA     AR 0F           * SECTOR LOCATION
827A D78180                    JSR     PREAD:SECTOR
827B 195F00                    LPI     $1F00  ,RD
827C A0080A                    IOR     PL < $00,CL
827D 8180CF                    TPA     AR 0C
827E 194F10                    LPI     $1F10
827F 0200E8    K827F           ORX     R0 < 00+,PL
8280 0208E0    K8280           ORX     PL < 00+,R0
8281 A01F0F                    SET     DUM < 0 ,RD
8282 FE6A18                    BNH     1,CH      LOOP
8283 FA6A08                    BNL     0,CH      LOOP
8284 FE6A0A                    BNH     0,CL      LOOP
8285 AC0888                    IADD    PL < 8,PL
8286 801FEF                    OR      DUM < 00+,00 ,RD
8287 FE6A4B                    BNH     4,CH      LOOP  * FIRST CHARACTER = '3'
8288 FA6A0B                    BNL     0,CH      LOOP
8289 5A6AC7                    BNR     CL+,R7    LOOP  * COMPARE NEXT
828A 801FEF                    OR      DUM < 00+,00 ,RD
828B 5A6AD6                    BNR     CH+,R6    LOOP  * AND THIRD
828C DA6AA5                    BNR     CL,R5     LOOP  * AND FOURTH
828D 801FEF    K828D           OR      DUM < 00+,00 ,RD
828E FE6A28                    BNH     2,CH      LOOP  * ALL OTHERS = SPACE
828F FA6A0B                    BNL     0,CH      LOOP
8290 FA8D08                    BNL     0,PL      K828D * REMAINING ENTRIES

8291 AC0020                    IADD    R0 < 2,R0
8292 0208E0                    ORX     PL < 00+,R0
8293 801F0F                    OR      DUM < R0,00  ,RD
8294 0208EA                    ORX     PL < 00+,CL
8295 8180FF                    TPA     AR 0F           * 1ST RECORD TO LOAD
8296 D78180    FETCH:RECORD    JSR     PREAD:SECTOR
8297 195F00                    LPI     $1F00  ,RD
8298 66D94D                    BTH     4,SH      DISP:PEDM * PARITY ERROR DM
8299 66E328                    BTH     2,CH      K82E3
829A 7F7888                    BNH     8,CH      PERR:D88
829B 995F04                    LPI     $1F04 ,RD
829C D2CFBF                    BER     CH,00     PRINT:RECORD
829D A80008                    IOR     R0 < 0,CH
829E 195F06                    LPI     $1F06 ,RD
829F 0208EA                    ORX     PL < 00+,CL
82A0 0180DF                    TPA     AR 0D           * STARTING ADDRESS
82A1 995F08                    LPI     $1F08 ,RD
82A2 A0D308                    IOR     R3 < 0,CH       * COUNT OF DATA
82A3 52D83F                    BER     R3,00     CHECK:PEDM * SEE IF ERRORS
82A4 0184CF                    TPA     AR 0C ,+2
82A5 7F7800                    BNH     0,R0      PERR:D88
82A6 62AB1C                    BTL     1,SL      K82AB
82A7 54D080                    JSR     CLEAR:DM
82A8 54D6B0                    JSR     TEST:CM         * Set CM = B00000
82A9 A00C1C                    IOR     SL < 1,SL
82AA 5EE080                    JMP     PLAY1+2
```

```
82AB 2B0130   K82AB          IAND    R1 < 3,R0
82AC 000C1C                  OR      SL < R1,SL
82AD 728C30                  SEL     3,R0        LD:CM * LOAD CONTROL MEM
82AE 7B7810                  BNL     1,R0        PERR:DBB
82AF 0B80DF   LD:CM          TAP     AR 0D
82B0 0380CF                  XPA     AR 0C
82B1 A0C2CF   K82B1          SET     R2 < $3C          * 60 MOD FOR DA FORMAT
82B2 0392CF   K82B2          XPA     AR 0C  ,+1 ,RD
82B3 66D940                  BTH     4,SH        DISP:PEDM
82B4 575EB0                  JSR     WRITE:DM:1BYTE
82B5 8382CF                  XPA     AR 0C ,+1
82B6 2FC3F3                  IADD    R3 < $FF,R3       * DEC COUNT
82B7 52D83F                  BER     R3,00       CHECK:PEDM
82B8 2FC2F2                  IADD    R2 < $FF,R2
82B9 5A822F                  BNR     R2,00       K82B2
82BA 9ACBEB                  ADCX    PL < 00+,PL ,CS * BUMP OVER 'BC' CHARACTER
82BB 0E8180                  JMP     K82B1

82BC 20423F   LD:CM          SET     R2 < $13
82BD 8B80CF                  TAP     AR 0C
82BE 801FEF                  OR      DUM < 00+,00 ,RD
82BF 001EDF                  OR      K < CH+,00 ,RD
82C0 0011DF                  OR      R1 < CH+,00 ,RD
82C1 A0000B                  IOR     R0 < 0,CH
82C2 8180CF                  TPA     AR 0C
82C3 0208E0                  ORX     PL < 00+,R0
82C4 8180DF                  TPA     AR 0D
82C5 0B80DF                  TAP     AR 0D             * ADDRESS TO WRITE TOO
82C6 8182DF                  TPA     AR 0D ,+1
82C7 54B580                  JSR     TEST:ONE:CM       * Write Read one location
82C8 0C833F                  S8C     R3 < R3,00 ,CC
82C9 52D83F                  BER     R3,00       CHECK:PEDM * LOAD FINISHED
82CA 0C822F                  S8C     R2 < R2,00 ,CC
82CB 62BD1D                  BTL     1,SH        LD:CM+1
82CC 8880CF                  TAP     AR 0C
82CD 0182CF                  TPA     AR 0C ,+1         * BUMP OVER '8C' CHAR
82CE 5E8CB0                  JMP     LD:CM

82CF 195F0A   PRINT:RECORD   LPI     $1F0A ,RD
82D0 548080                  JSR     SELECTCRT
82D1 000FEF                  OR      DUM < 00+,00
82D2 001EDF                  OR      K < CH+,00 ,RD
82D3 D41180                  JSR     OUTPUTK
82D4 FED248                  BNH     4,PL        .-2
82D5 7AD278                  BNL     7,PL        .-3
82D6 D42480                  JSR     CRLF
82D7 547CB0                  JSR     SELECTAR1AL
82D8 6E9640   CHECK:PEDM     BFH     $4,SH       FETCH:RECORD * NEXT IF NO PEDMS'
82D9 5480B0   DISP:PEDM      JSR     SELECTCRT
82DA A14E0F                  SET     K < $50
82DB D41180                  JSR     OUTPUTK           * 'P'
82DC 196F00                  LPI     $1F00 ,W1
82DD A01F0F                  SET     DUM < 0 ,RD
82DE 2ACDFD   PLAY1          IAND    SH < $8F,SH
82DF 547CB0                  JSR     SELECTAR1AL
82E0 0380FF                  XPA     AR 0F
82E1 03C2FF                  XPA     AR 0F ,-1
82E2 DE9680                  JMP     FETCH:RECORD      * RETRY

82E3 EAF22C   K82E3          8FL     2,SL        K82F2
```

```
82E4 D4ED80                     JSR     CHECKSUMS
82E5 D4E280                     JSR     SCAN:DM
82E6 198001                     LPI     $2001
82E7 8181EF                     TPA     AR 1E
82E8 190000                     LPI     0
82E9 0181FF                     TPA     AR 1F
82EA 280C4C                     IAND    SL < $4,SL
82EB 6AED4C                     BFL     4,SL      .+2
82EC 200CCF                     SET     SL < $C
82ED DC0300                     JMP     START:ADDRESS

82EE 200F0F                     SET     DUM < 0
82EF 200F0F                     SET     DUM < 0
82F0 200F0F                     SET     DUM < 0
82F1 5F4D80                     JMP     CALC:PARITY

82F2 280C4C   K82F2             IAND    SL < 4,SL
82F3 548080                     JSR     SELECTCRT
82F4 991000                     LPI     0 ,R0
82F5 FEF7F8                     BNH     $F,CH     .+2
82F6 F258FB                     BEL     $F,CH     K825B
82F7 001EDF                     OR      K < CH+,00 ,R0
82F8 D41180                     JSR     OUTPUTK
82F9 DEF580                     JMP     K82F2+3

              *
              *       INPUTC takes data from the input device and verifies
              *       data.  If the first character is called for, the
              *       system checks for an '2'. If more than 4 characters are
              *       typed, the system hangs here till an SF key is
              *       pressed. Any key depressed other than a SF key
              *       causes bit 08 of the SL to be set.

82FA 200E1F   INPUTC            SET     K < 1           * DEVICE TO INPUT FROM
82FB D48180                     JSR     SELECTK8
82FC D7BD80                     JSR     WAITONINPUT     * HANG TILL DATA IN
82FD A0000E                     IOR     R0 < 00,K
82FE 548080                     JSR     SELECTCRT
82FF A00E00                     IOR     K < 00,R0
8300 E30A4D                     BTL     4,SH      SF:DEPRESSED
8301 76FA00                     BEH     0,R0      INPUTC * ILLEGAL
8302 66FA80                     BTH     8,R0      INPUTC * > 7F
8303 66FA1C                     BTH     1,SL      INPUTC * 4TH CHAR
8304 60118C                     BTL     8,SL      OUTPUTK
8305 210E0F                     SET     K < $40         * LOAD IN '2'
8306 5AFA0E                     BNR     R0,K      INPUTC * 1ST CHAR MUST BE 2
8307 D41180                     JSR     OUTPUTK
8308 A00C8C                     IOR     SL < $8,SL      * INFORM OF MORE THAN ONE
8309 DEFA80                     JMP     INPUTC

830A 63418C   SF:DEPRESSED      BTL     8,SL      SF:NOKEYS * No other keys have been pressed
830B 8208EE                     ORX     PL < 00+,K      * TRANSFER KEY TO PL
830C 8181DF                     TPA     AR 1D
830D EB1B2C                     BFL     2,SL      K831B
830E 7F130E                     BNH     0,K       K8313
830F F359FE                     BEL     $F,K      SF15:DEPRESSED
8310 E2FA6E                     BTL     6,K       INPUTC * CANNOT BE '6 OR '7
8311 21070F                     SET     R7 < $40        * PRELOAD 2
8312 DF1980                     JMP     K8319
8313 FEFA1E   K8313             BNH     1,K       INPUTC
```

```
8314 A1070E                    IOR     R7 ( $40,K
8315 A8C7C7                    IAND    R7 ( $FC,R7
8316 7B18C7                    BNL     $C,R7     K8318
8317 A00C4C                    IOR     SL ( 4,SL
8318 A80E3E    K8318           IAND    K ( 3,K
8319 574680    K8319           JSR     FORM:BOOT:DISK
831A DF2680                    JMP     K8326

831B AD0710    K8318           IADD    R7 ( $41,R0
831C 6B264C                    BFL     4,SL      K8326
831D A1071F                    SET     R7 ( $41        * 2A
831E 53260F                    BER     R0,00     K8326
831F A00E3F                    SET     K ( 3
8320 C8250E                    BLER    R0,K      K8325
8321 A00E6F                    SET     K ( 6
8322 4B240E                    BLER    R0,K      K8324
8323 AC0717                    IADD    R7 ( 1,R7
8324 AC0717    K8324           IADD    R7 ( 1,R7
8325 AC0717    K8325           IADD    R7 ( 1,R7
8326 A1090F    K8326           SET     PH ( $40
8327 200807                    IOR     PL ( 0,R7
8328 D42D80                    JSR     OUTPHPL

8329 19802F    K8329           LPI     $202F           * '
832A D42D80                    JSR     OUTPHPL
832B 0881AF                    TAP     AR 1A           * GET DEVICE
832C A0C209                    IOR     R2 ( $30,PH
832D A0C19F                    SET     R1 ( $39
832E 542080                    JSR     HEXADJUST
832F D41880                    JSR     OUT2PL
8330 998021                    LPI     $2021
8331 542A80                    JSR     REPEATOUTK
8332 A08E8F                    SET     K ( $28         * (
8333 98024F                    LPI     $424F           * 'BO'
8334 542C80                    JSR     OUTKPHPL
8335 984453                    LPI     $5453           * 'OTS'
8336 542C80                    JSR     OUTKPHPL
8337 214E4F                    SET     K ( $54
8338 984241                    LPI     $5241           * 'TRA'
8339 542C80                    JSR     OUTKPHPL
833A 9B403D                    LPI     $503D           * 'P-'
833B D42D80                    JSR     OUTPHPL
833C 214E2F                    SET     K ( $52
833D 99C329                    LPI     $3329           * 'R3)'
833E 542C80                    JSR     OUTKPHPL
833F D42480                    JSR     CRLF
8340 5E7380                    JMP     LOAD:PROGRAM

            *
            *       Pressing a SF key without entering data characters
            *       vectors to this location.

8341 7EFA0E    SF:NOKEYS       BNH     0,K       INPUTC
8342 F359FE                    BEL     $F,K      SF15:DEPRESSED
8343 E2FA6E                    BTL     6,K       INPUTC
8344 5746B0                    JSR     FORM:BOOT:DISK
8345 DF2980                    JMP     K8329

            *       Forms the address of the disk from the SF key.
            *       Returns 310,320,330 or 810,820,830 or their
```

```
                *       second unit counterparts.

8346 3C088E     FORM:BOOT:DISK      IMUL ALBL PL < 8,K        * SF KEY * 8
8347 A80980                         IAND      PH < 8,PL
8348 29C808                         IAND      PL < $70,PL
8349 2C0939                         IADD      PH < 3,PH
834A AC4808                         IADD      PL < $10,PL
834B 0181AF                         TPA       AR 1A
834C B7800F                         RTS

                *       Calculates Parity of Control Memory word in K, PH and PL.
                *       Returns back R0 = FF if bad, or R0 = 0 if good.

834D 0200E8     CALC:PARITY         ORX       R0 < 00+,PL
834E 84011E                         XOR       R1 < R1,K
834F 040110                         XOR       R1 < R1,R0
8350 04401F                         SDC  AHBL R0 < R1,00
8351 040101                         XOR       R1 < R0,R1
8352 8C0041                         IMUL ALBL R0 < 4,R1
8353 040101                         XOR       R1 < R0,R1
8354 6B5641                         BFL       4,R1      .+2
8355 240181                         IXOR      R1 < 8,R1
8356 E0E881                         BTL       8,R1      CLEAR:R0:EXIT
8357 23C0FF                         SET       R0 < $FF        * BAD PARITY
8358 87800F                         RTS

8359 D42480     SF15:DEPRESSED      JSR       CRLF
835A 99000D                         LPI       $000D
835B 54C880                         JSR       REWRITERESETV
835C A8CDFD                         IAND      SH < $3F,SH
835D 5C0E80                         JMP       SF'15RESETVECT

                *       Writes one byte to DM at PHPL  from CH.
                *       Reads data back and checks for Parity and Bit errors

835E 2D0E08     WRITE:DM:1BYTE      IOR       K < 0,CH
835F 202F0E                         IOR       DUM < 0,K ,W1
8360 A01F0F                         SET       DUM < 0 ,RD
8361 67634D                         BTH       4,SH      .+2
8362 D0A68E                         BER       CH,K      CLEAN:EXIT
8363 0406BE                         XOR       R6 < CH,K
8364 0A04EE                         ANDX      R4 < 00+,K
8365 A1014F                         SET       R1 < $44        * 'D'
8366 A1432F                         SET       R3 < $52        * 'R'
8367 DD2D80                         JMP       SHOW:ERROR      * REDM

                *       Routine writes two bytes, R1 and R0 to DM at location
                *       pointed to by PHPL.  The Bytes are then read back and
                *       checked for parity and bit errors.  The routine will
                *       return only if no errors.

8368 202F01     WRITE:DM:2BYTES     IOR       DUM < 0,R1 ,W1
8369 203F00                         IOR       DUM < 0,R0 ,W2
836A A01F0F     READ:DM:2BYTES      SET       DUM < 0 ,RD
836B E76E4D                         BTH       4,SH      .+3   * PARITY ERROR
836C DB6E81                         BNR       CH,R1     .+2
836D D0A6A0                         BER       CL,R0     CLEAN:EXIT
836E 0604A0                         XORX      R4 < CL,R0
836F 20060F                         SET       R6 < 0
8370 A1014F                         SET       R1 < $44        * 'D'
```

```
8371 5CC680                     JMP     SET8ITERROR     * 8EDM

8372 A08000     OR20WITHR0      IOR     R0 ( $20,R0
8373 87800F                     RTS

8374 A20000     OR80WITHR0      IOR     R0 ( $80,R0
8375 87800F                     RTS

8376 22002F     PERR:D82 SET    R0 ( $82
8377 DF7980                     JMP     DISKERROR

8378 22008F     PERR:D88 SET    R0 ( $88
8379 A00800     DISKERROR       IOR     PL ( 0,R0
837A 20090F                     SET     PH ( 0
8378 21034F                     SET     R3 ( $44        * 'DISK'
837C 21029F                     SET     R2 ( $49
837D A1413F                     SET     R1 ( $53
837E 21008F                     SET     R0 ( $48
837F 543480                     JSR     DISPLAY:ERROR+3
8380 5E3780                     JMP     K8237
                *       Reads one sector from disk at sector pointed to by AR 0F.
                *       Data from sector is transferred to DM starting at $1F00.
                *       All error handling is performed by this routine.


8381 579880     PREAD:SECTOR    JSR     PDISK:OPEN
8382 20000F                     SET     R0 ( 0
8383 57A780                     JSR     TXFR:SECTOR
8384 58810F                     BNR     R0,00     PREAD:SECTOR
8385 D7C080                     JSR     CHECKSTAT1
8386 D41280                     JSR     FORCEOUT[K]
8387 57CB80                     JSR     CHECKSTAT2
8388 994F00                     LPI     $1F00
8389 A0020F                     SET     R2 ( 00
838A D78D80     K838A           JSR     WAITONINPUT
838B 98822E                     ADC     R2 ( R2,K ,CC
838C 20010E                     IOR     R1 ( 0,K
838D D78D80                     JSR     WAITONINPUT
838E 98822E                     ADC     R2 ( R2,K ,CC
838F A0000E                     IOR     R0 ( 0,K
8390 576880                     JSR     WRITE:DM:2BYTES
8391 000FEF                     OR      DUM ( 00+,00
8392 0000EF                     OR      R0 ( 00+,00
8393 588A08                     BNR     R0,PL     K838A
8394 D78D80                     JSR     WAITONINPUT     * GET CHECKSUM
8395 D0A62E                     BER     R2,K      CLEAN:EXIT
8396 A2407F     PERR:I97 SET    R0 ( $97
8397 DF7980                     JMP     DISKERROR

                *       Waits for selected disk to come ready.  Sets AB bus
                *       to opening sequence and sends an '00' OBS. Disk will
                *       respond with 'C0' if Dumb, or 'D0' if intelligent.
                *       R2 = 0 on return if Dumb disk answered. R2 = 1 if
                *       an intelligent disk answered. If neither answered, an
                *       I90 will be issued.

8398 E8988D     PDISK:OPEN      BFL     8,SH      PDISK:OPEN
8399 A00D2D                     IOR     SH ( 2,SH
839A A28E0F                     SET     K ( $A0
8398 978800                     CIO     CA8
```

```
839C A00E0F                     SET     K < 0
839D D41180                     JSR     OUTPUTK
839E 57D680                     JSR     PTIMED:INPUT    * WAIT FOR RESPONSE
839F E39A1D                     BTL     1,SH    PDISK:OPEN+2 * NO RESP
83A0 A0020F                     SET     R2 < 0
83A1 F7A4CE                     BEH     $C,K    K83A4 * DUMB DISK
83A2 7FA5DE                     BNH     $D,K    PERR:I90
83A3 20021F                     SET     R2 < 1          * INTELLIGENT DISK
83A4 87800F   K83A4             RTS

83A5 22400F   PERR:I90 SET      R0 < $90
83A6 DF7980                     JMP     DISKERROR

              *       Routine transfer to the disk controller the Command byte
              *       and the requested sector.  R2 determines if an Dumb or
              *       Intelligent disk is being addresses. If an echo fails,
              *       R0 will be non-zero

83A7 210E0F   TXFR:SECTOR       SET     K < $40
83A8 979800                     CID     CAB
83A9 0881AF                     TAP     AR 1A
83AA F3AD39                     BEL     3,PH    PFIXED:DISK
83AB A04000                     IOR     R0 < $10,R0
83AC 0181AF                     TPA     AR 1A
83AD 8880FF   PFIXED:DISK       TAP     AR 0F
83AE 0182FF                     TPA     AR 0F ,+1
83AF D7DD80                     JSR     OUT:IN          * ECHO BYTE
83B0 D88F0F                     BNR     R0,00   PDISK:OPEN:EXIT
83B1 68B712                     BFL     1,R2    DUMB:DISK
83B2 20000F                     SET     R0 < 0
83B3 D7DD80                     JSR     OUT:IN
83B4 D88F0F                     BNR     R0,00   PDISK:OPEN:EXIT
83B5 200009                     IOR     R0 < 0,PH
83B6 5FB880                     JMP     DUMB:DISK+1
83B7 29C0F9   DUMB:DISK         IAND    R0 < $7F,PH
83B8 D7DD80                     JSR     OUT:IN
83B9 D88F0F                     BNR     R0,00   PDISK:OPEN:EXIT
83BA A00008                     IOR     R0 < 0,PL
83BB D7DD80                     JSR     OUT:IN
83BC 53BF0F                     BER     R0,00   PDISK:OPEN:EXIT

83BD 28CD8D   WAITONINPUT       IAND    SH < $FB,SH
83BE 6BBE2D                     BFL     2,SH    .
83BF 87800F   PDISK:OPEN:EXIT   RTS

              *       Bit tests the first status byte received from the disk

83C0 D7BD80   CHECKSTAT1        JSR     WAITONINPUT
83C1 E3C51E                     BTL     1,K     PERR:I9B
83C2 63C72E                     BTL     2,K     PERR:I91
83C3 E3C94E                     BTL     4,K     PERR:I94
83C4 DC8380                     JMP     PDELAY2

83C5 A2408F   PERR:I98 SET      R0 < $9B
83C6 DF7980                     JMP     DISKERROR
83C7 A2401F   PERR:I91 SET      R0 < $91
83C8 DF7980                     JMP     DISKERROR
83C9 A2404F   PERR:I94 SET      R0 < $94
83CA DF7980                     JMP     DISKERROR
```

```
                *       Bit tests the second status byte received from the disk

83CB D7BD80     CHECKSTAT2      JSR     WAITONINPUT
83CC E3D41E                     BTL     1,K     PERR:I95
83CD 63D02E                     BTL     2,K     PERR:I93
83CE E3D24E                     BTL     4,K     PERR:I96
83CF 87B00F                     RTS

83D0 22403F     PERR:I93 SET    R0 < $93
83D1 DF7980                     JMP     DISKERROR
83D2 22406F     PERR:I96 SET    R0 < $96
83D3 DF7980                     JMP     DISKERROR
83D4 22405F     PERR:I95 SET    R0 < $95
83D5 DF7980                     JMP     DISKERROR

                *       This routine will wait for an input from IO device.
                *       The routine exits if a character is received
                *       or if the timeout expires.  On return, if the Carry is
                *       clear, a character was received, else timeout

83D6 9B0B01     PTIMED:INPUT    LPI     $4B01           * SET DELAY TIME
83D7 ABCDCD                     IAND    SH < $FC,SH
83D8 63DC2D                     BTL     2,SH    .+4     * GOT DATA, EXIT
83D9 9AC8E8                     ADCX    PL < 00+,PL ,CS
83DA 6BD81D                     BFL     1,SH    PTIMED:INPUT+2
83DB A00D2D                     IOR     SH < 2,SH
83DC 87800F                     RTS

                *       Sends a byte from R0 to the selected device and waits
                *       waits for an echo.  If the same character is sent back,
                *       R0 will be cleared. R0 <> 0 then echo error.

83DD A00D2D     OUT:IN          IOR     SH < 2,SH
83DE A00E00                     IOR     K < 0,R0
83DF D41180     PEXIT1          JSR     OUTPUTK
83E0 D78D80                     JSR     WAITONINPUT
83E1 84D00E                     XOR     R0 < R0,K       * SEE RESPONSE
83E2 5C8280                     JMP     PDELAY1

                *       The location pointed to by the PHPL pair is read.  The
                *       data from the Control Memory read is then stored in all
                *       CM locations from 000E to 0000.
                *

83E3 D4AF80     WRITE1ST16CM    JSR     READ:CM
83E4 81800F                     TPA     AR 00
83E5 99000E                     LPI     $E
83E6 01C21F                     TPA     AR 01   ,-1
83E7 D4B1B0                     JSR     WRITE:CM
83E8 A7CEFE                     IXOR    K < $FF,K       * NORMAL -1
83E9 0B801F                     TAP     AR 01
83EA 77E609                     BEH     0,PH    WRITE1ST16CM+3
83EB 87800F                     RTS

                *       Display the SL register, (Memory Bank Selects)

83EC AB880C     DISP:BANK       IAND    PL < $E0,SL
83ED D41B80                     JSR     OUT2PL
83EE A08EEF                     SET     K < $2E         * '.'
83EF 5C1180                     JMP     OUTPUTK
```

```
                *
                *       NULLS ARE BLANK LOCATIONS HERE
                *
83F0 800000             OR      R0 < R0,R0
83F1 800000             OR      R0 < R0,R0
83F2 800000             OR      R0 < R0,R0
83F3 800000             OR      R0 < R0,R0
83F4 800000             OR      R0 < R0,R0
83F5 800000             OR      R0 < R0,R0
83F6 800000             OR      R0 < R0,R0
83F7 800000             OR      R0 < R0,R0
83F8 800000             OR      R0 < R0,R0
83F9 800000             OR      R0 < R0,R0


83FA DE4D80     K83FA           JMP     KEY:SF+I
83FB 5E7380     K83FB           JMP     LOAD:PROGRAM
83FC 5F4D80     K83FC           JMP     CALC:PARITY
83FD 5CB880     VSIZE:CM        JMP     SIZE:CM
83FE 800000     CHECK:LOC       FCB     0
83FF 800000                     FCB     0
```

Symbol Dump
Alphanumeric Symbol Sort

```
$808D BEGIN:SIZER     $834D CALC:PARITY     $80A2 CHECK:DONE      $83FE*CHECK:LOC       $8208 CHECK:PEDM      $809A CHECK:RESULT
$83C0 CHECKSTAT1      $83CB CHECKSTAT2      $80E0 CHECKSUMS       $80A6 CLEAN:EXIT      $80D0 CLEAR:DM        $80E8 CLEAR:R0:EXIT
$80C3 DM:BITERROR     $8024 CRLF            $80A7 DELAY:2000      $80A8 DELAY:[PHPL]                          $8379 DISKERROR
$83EC DISP:BANK       $82D9 DISP:PEDM       $8128 DISP:VEDM       $812F DISP:VEDM       $812C DISP:VExM       $8064 DISPLAY:DM
$8031 DISPLAY:ERROR                         $8016 DISPLAY:PHPL                          $83B7 DUMB:DISK       $800E*ERR:CHECKDM
$800F*ERR:CHECKDM     $8002 ERR:DATA:DM     $8000*ERR:PARITY      $823A ERROR           $8296 FETCH:RECORD
$8012 FORCEOUT[K]     $8346 FORM:BOOT:DISK                        $0001 HARD:RESET      $8020 HEXADJUST       $82FA INPUTC
$8039 K8039           $805D K805D           $8069 K8069           $806F K806F           $80F5 K80F5           $80FA K80FA
$8129 K8129           $8155 K8155           $81A8 K81A8           $81AD K81AD           $B1F3 K81F3           $8212 K8212
$8213 K8213           $8216 K8216           $821B K8218           $821D K821D           $8237 K8237           $825B K8258
$827F K827F           $8280 K8280           $828D K828D           $82AB K82AB           $82B1 K82B1           $8282 K8282
$82E3 K82E3           $82F2 K82F2           $8313 K8313           $8318 K8318           $8319 K8319           $8318 K831B
$8324 K8324           $8325 K8325           $8326 K8326           $8329 K8329           $838A K838A           $83A4 K83A4
$83FA*K83FA           $83FB*K83FB           $83FC*K83FC           $824C KEY:SF          $828C LO:DM           $82AF*LD:DM
$8273 LOAD:PROGRAM                          $826A LOOP            $8090 MEM:PRESENT     $808F NEXT:LOC        $8104 NOPARITY:ERR
$8022 NOT:A:HEX       $801E*NULL:CR         $807A OR01WITHR0      $80AC OR04WITHR0      $8372 OR20WITHR0      $8374 OR80WITHR0
$8018 OUT2PL          $8019 OUT2R0          $8015*OUT4            $83DD OUT:IN          $802C OUTKPHPL        $802D OUTPHPL
$8011 OUTPUTK         $8010 OUTPUTSPACE     $800C PARITY:TEST     $8141 PASSED:PARITY                         $8082 PDELAY1
$8083 PDELAY2         $8084 PDELAY3         $8085 PDELAY4         $83BF PDISK:OPEN:EXIT                       $8398 PDISK:OPEN
$822E PE:IN:DM        $8227 PE:IN:DM        $8376 PERR:082        $8378 PERR:088        $83A5 PERR:190        $83C7 PERR:191
$83D0 PERR:193        $83C9 PERR:194        $83D4 PERR:195        $83D2 PERR:196        $8396*PERR:197        $83C5 PERR:198
$830F*PEXIT1          $83A0 PFIXED:DISK     $82DE PLAY1           $8003 POWERON         $8131 POWERON:VECT
$8381 PREAD:SECTOR                          $8238 PRESS:RESET     $82CF PRINT:RECORD                          $80F6 PROM:CHECKS
$8306 PTIMED:INPUT                          $80AE*ROM:DELAYED     $80AF READ:DM         $836A READ:DM:2BYTES
$802A REPEATOUTK      $8001*RESETV          $80CC*REWRITE'15VEC                         $80CB REWRITERESETV
$8071 RIGHT:PAREN     $80E2 SCAN:DM         $807C SELECTARIAL     $8080 SELECTCRT       $8081 SELECTK8        $8028 SEND:PH
$8004 SERIAL#         $80C6 SET8ITERROR     $000E SF'15RESETVECT                        $8359 SF15:DEPRESSED
$830A SF:DEPRESSED                          $8341 SF:NOKEYS       $812D SHOW:ERROR      $0088 SIZE:DM         $8096 SIZE:DM0
$808A SIZE:DM         $80A3 SIZE:FOUND      $0003 START:ADDRESS                         $820A TEST:DM:DM      $80D6 TEST:DM
$80D7 TEST:DM:[PHPL]                        $80DD TEST:NEXT:DM                          $80E5 TEST:NEXT:DM
$8085 TEST:ONE:DM     $83A7 TXFR:SECTOR     $83FD VSIZE:DM        $800D VSIZE:DM        $838D WAITONINPUT     $83E3 WRITE1ST16DM
$8081 WRITE:DM        $835E WRITE:DM:1BYTE                        $8368 WRITE:DM:2BYTES
```

Symbol Dump
Numeric Symbol Sort

```
$0001 HARD:RESET      $0003 START:ADDRESS                $000E SF'15RESETVECT               $8000*ERR:PARITY
$8001*RESETV          $8002 ERR:DATA:DM   $8003 POWERON  $8004 SERIAL#   $800C PARITY:TEST  $800D VSIZE:DM
$800E*ERR:CHECKDM     $800F*ERR:CHECKDM   $8010 OUTPUTSPACE $8011 OUTPUTK $8012 FORCEOUT[K] $8015*OUT4
$8016 DISPLAY:PHPL                        $8018 OUT2PL    $8019 OUT2R0    $801E*NULL:CR      $8020 HEXADJUST
$8022 NOT:A:HEX       $8024 CRLF          $802B SEND:PH   $802A REPEATOUTK $802C OUTKPHPL     $802D OUTPHPL
$8031 DISPLAY:ERROR                       $8039 K8039     $805D K805D     $8064 DISPLAY:DM   $8069 K8069
$806F K806F           $8071 RIGHT:PAREN   $807A OR01WITHR0 $807C SELECTARIAL $8080 SELECTCRT  $8081 SELECTK8
$8082 PDELAY1         $8083 PDELAY2       $8084 PDELAY3   $8085 PDELAY4   $8088 SIZE:DM      $808A SIZE:DM
$808D BEGIN:SIZER     $808F NEXT:LOC      $8096 SIZE:DM0  $809A CHECK:RESULT                 $809D MEM:PRESENT
$80A2 CHECK:DONE      $80A3 SIZE:FOUND    $80A6 CLEAN:EXIT $80A7 DELAY:2000 $80A8 DELAY:[PHPL]
$80AC OR04WITHR0      $80AE*ROM:DELAYED   $80AF READ:DM   $80B1 WRITE:DM   $80B5 TEST:ONE:DM $80C3 DM:81TERROR
$80C6 SET81TERROR     $80C8 REWRITERESETV                $80CC*REWRITE'15VEC                $80D0 CLEAR:DM
$80D6 TEST:DM         $80D7 TEST:DM:[PHPL]               $80DD TEST:NEXT:DM                 $80E2 SCAN:DM
$80E5 TEST:NEXT:DM                        $80E8 CLEAR:R0:EXIT          $80ED CHECKSUMS     $80F5 K80F5
$80F6 PROM:CHECKS     $80FA K80FA         $8104 NOPARITY:ERR           $8129 K8129         $812B DISP:VECM
$812C DISP:VExM       $812D SHOW:ERROR    $812F DISP:VEDM  $8131 POWERON:VECT              $8141 PASSED:PARITY
$8155 K8155           $81A8 K81A8         $81AD K81A0     $81F3 K81F3     $820A TEST:DM:DM   $8212 K8212
$8213 K8213           $8216 K8216         $8218 K8218     $821D K821D     $8227 PE:IN:DM     $822E PE:IN:DM
$8237 K8237           $823A ERROR         $823B PRESS:RESET $824C KEY:SF   $8258 K825B       $826A LOOP
$8273 LOAD:PROGRAM                        $827F K827F     $8280 K8280     $828D K8280        $8296 FETCH:RECORD
$82AB K82AB           $82AF*LD:DM         $82B1 K8281     $82B2 K82B2     $82BC LD:DM        $82CF PRINT:RECORD
$82D8 CHECK:PEDM      $82D9 DISP:PEDM     $82DE PLAY1     $82E3 K82E3     $82F2 K82F2        $82FA INPUTC
$830A SF:DEPRESSED                        $8313 K8313     $8318 K8318     $8319 K8319        $831B K831B
$8324 K8324           $8325 K8325         $8326 K8326     $8329 K8329     $8341 SF:NOKEYS    $8346 FORM:BOOT:DISK
$834D CALC:PARITY     $8359 SF15:DEPRESSED               $835E WRITE:DM:18YTE               $8368 WRITE:DM:2BYTES
$836A READ:DM:2BYTES                      $8372 OR20WITHR0 $8374 OR80WITHR0 $8376 PERR:D82    $837B PERR:D88
$8379 DISKERROR       $8381 PREAD:SECTOR                 $838A K838A     $8396*PERR:I97     $8398 PDISK:OPEN
$83A4 K83A4           $83A5 PERR:I98      $83A7 TXFR:SECTOR $83AD PFIXED:DISK $83B7 DUMB:DISK $83B0 WAITONINPUT
$83BF PDISK:OPEN:EXIT                     $83C0 CHECKSTAT1 $83C5 PERR:I98 $83C7 PERR:I91     $83C9 PERR:I94
$83CB CHECKSTAT2      $83D0 PERR:I93      $83D2 PERR:I96  $83D4 PERR:I95   $83D6 PTIMED:INPUT
$83DD OUT:IN          $83DF*PEXIT1        $83E3 WRITE1ST16DM           $83EC DISP:BANK     $83FA*K83FA
$83F8*K83FB           $83FC*K83FC         $83FD VSIZE:DM  $83FE*CHECK:LOC
```