```
                TITLE   Infocom INTERLOGIC interpreter disassembly, 5/27/84

        ; ***************************************************************************
        ; *                                                                         *
        ; *                 Infocom INTERLOGIC interpreter disassembly              *
        ; *                      Apple II/6502 version, release 3                   *
        ; *                    As used in interactive fiction games                 *
        ; *                                                                         *
        ; *       The INTERLOGIC interpreter is copyrighted by Infocom, Inc.        *
        ; *                                                                         *
        ; *   This disassembly and the comments therof are copyright (C) 1984 by    *
        ; *                        Eric L. Smith                                    *
        ; *                        230 South 500 West Suite 133                     *
        ; *                        Salt Lake City, Utah  84101                      *
        ; *                        (801) 582-3371                                   *
        ; *                                                                         *
        ; * This disassembly represents well over 300 hours of intense study.  It   *
        ; * is intended for private, noncommercial use only.  Any comments or       *
        ; * questions about it should be addressed to the above address.  There is  *
        ; * no warranty, express or implied, as to the accuracy of this disassembly *
        ; * or its fitness for any particular purpose.  I assume no liability for   *
        ; * any damages, actual or alleged, direct or indirect, resulting from the  *
        ; * use of, or inability to use this disassembly.                           *
        ; *                                                                         *
        ; ***************************************************************************

                PAGE
```

```


                                .6502
                                .SALL
                                .SFCOND

 0000                   VERSN   EQU     0                       ; 0 is old version, 1 is new
 0000                   RNGDBG  EQU     0                       ; RNG debug
 0001                   LC40    EQU     1                       ; 40 column lower case

                        ; define memory usage

 0100                   LDORG   EQU     $0100                   ; where to load

 007F                   ZPORG   EQU     $7F                     ; origin of zero page usage
 0200                   BUFFER  EQU     $0200                   ; I/O buffer
 00E0                   STCKMX  EQU     $E0                     ; maximum size of stack in words
 03E8                   STCKLC  EQU     $03E8                   ; base address of stack (works down)
 0228                   STKLIM  EQU     STCKLC-2*STCKMX         ; lower limit of stack

 0779                   PRTWDT  EQU     $0779                   ; printer carriage width

                                IFF     VERSN
 0800                   MAINOR  EQU     $0800                   ; origin of main program
 2200                   VMTORG  EQU     MAINOR+$1A00            ; origin of virtual memory tables
 2400                   RWTSOR  EQU     VMTORG+$0200            ; origin of RWTS routines
 2C00                   FIRFLC  EQU     RWTSOR+$0800            ; first location available
 BFFF                   LSTFLC  EQU     $C000-1                 ; last potential location available
                                ENDIF

 2200                   VMT1LC  EQU     VMTORG+$0000            ; virtual memory page tables
 2280                   VMT2LC  EQU     VMTORG+$0080
 2300                   VMT3LC  EQU     VMTORG+$0100
 2380                   VMT4LC  EQU     VMTORG+$0180

 2900                   RWTS    EQU     RWTSOR+$0500            ; entry point of RWTS routines


                        ; Control characters

 000D                   CRCHAR  EQU     $0D                     ; carriage return
 000A                   LFCHAR  EQU     $0A                     ; line feed
 0009                   TBCHAR  EQU     $09                     ; horizontal tab
 000C                   FFCHAR  EQU     $0C                     ; form feed


                        ; Apple monitor ROM's zero page locations

 0020                   WNDLFT  EQU     $20                     ; screen window parameters
 0021                   WNDWDT  EQU     $21
 0022                   WNDTOP  EQU     $22
 0023                   WNDBOT  EQU     $23

 0024                   CURSRH  EQU     $24                     ; cursor position
 0025                   CURSRV  EQU     $25
```

```

0032             INVFLG  EQU     $32             ; inverse video output flag

0033             PROMPT  EQU     $33             ; line input prompt

0036             CSWL    EQU     $36             ; character output vector

004E             RNDLOC  EQU     $4E             ; location randomized by keyboard input


                 ; Apple monitor routines

FC22             VTAB    EQU     $FC22           ; adjust video pointer after cursor move
FC58             HOME    EQU     $FC58           ; clear screen window
FC9C             CLREOL  EQU     $FC9C           ; clear to end of line
FD0C             RDKEY   EQU     $FD0C           ; get a key from keyboard
FD6F             GETLN1  EQU     $FD6F           ; get a line from keyboard
FDED             COUT    EQU     $FDED           ; output a char to current device
FDF0             COUT1   EQU     $FDF0           ; output a char to screen

                         IFT     RNGDBG
                         ENDIF

                         PAGE
```

```

                        ; define our own zero page usage

 0000'          D                DSECT
                D                ORG      ZPORG
                D
 007F           D   SECPTK   DS       1                 ; number of sectors per track on disk
                D
 0080           D   OPCODE   DS       1                 ; opcode of current instruction
 0081           D   ARGCNT   DS       1                 ; instruction arguments
                D
 0082           D   ARG1     DS       2
 0084           D   ARG2     DS       2
 0086           D   ARG3     DS       2
 0088           D   ARG4     DS       2
                D
 008A           D   PRGIDX   DS       1                 ; PC low byte, index into page
 008B           D   PRGLPG   DS       2                 ; PC logical page number
 008D           D   PRGMPT   DS       2                 ; PC mem loc of logical page
 008F           D   PRGUPD   DS       1                 ; PC new page flag
 0090           D   PRGPPG   DS       1                 ; PC physical page number
                D
 0091           D   AUXLPG   DS       2                 ; AUX logical page number
 0093           D   AUXIDX   DS       1                 ; AUX low byte, index into page
 0094           D   AUXMPT   DS       2                 ; AUX mem loc of logical page
 0096           D   AUXUPD   DS       1                 ; AUX new page flag
 0097           D   AUXPPG   DS       1                 ; AUX physical page number
                D
 0098           D   GLBVAR   DS       2                 ; pointer to global variables
 009A           D   LOCVAR   DS       30                ; storage of local variables
                D
 00B8           D   SWPMEM   DS       2                 ; address of first swappable page
 00BA           D   FRZMEM   DS       2                 ; address of first frozen page
 00BC           D   FRZPGS   DS       1                 ; number of frozen pages
 00BD           D   SWPPGS   DS       1                 ; number of swappable phys. pages
                D
 00BE           D   MRUPAG   DS       1                 ; phys. pg. # of most recently used page
 00BF           D   LRUPAG   DS       1                 ; phys. pg. # of least recently used page
                D
 00C0           D   VMTAB1   DS       2                 ; virtual memory table pointers
 00C2           D   VMTAB2   DS       2
 00C4           D   VMTAB3   DS       2
 00C6           D   VMTAB4   DS       2
                D
 00C8           D   STKCNT   DS       1                 ; # items on stack
 00C9           D   STKPNT   DS       2                 ; stack pointer
 00CB           D   STKPSV   DS       2                 ; stack ptr save during call
 00CD           D   STKCSV   DS       1                 ; stack cnt save during call
                D
 00CE           D   TMPMOD   DS       1                 ; string output temporary char. mode
 00CF           D   PRMMOD   DS       1                 ; string output perm. char. mode
 00D0           D   PNYBCN   DS       1                 ; string output nybble counter
 00D1           D   PNYBBF   DS       2                 ; string output nybble buffer
                D
 00D3           D   INWORD   DS       6                 ; word to be packed
```

```
                          D
        00D9              D  LD9     DS     1
                          D
        00DA              D  PKWORD  DS     4                  ; packed word
                          D
        00DE              D  LDE     DS     1
        00DF              D  LDF     DS     1
        00E0              D  LE0     DS     1
        00E1              D  LE1     DS     1
                          D
        00E2              D  SBWDPT  DS     2
                          D
        00E4              D  ACB     DS     2
        00E6              D  ACC     DS     2
        00E8              D  ACD     DS     2
                          D
        00EA              D  MDFLAG  DS     1                  ; negative arg count for mult/div
                          D
        00EB              D  CHRPTR  DS     1                  ; char out buffer pointer
        00EC              D  CHRPT2  DS     1                  ; char out buffer pointer 2
        00ED              D  LINCNT  DS     1                  ; output line counter
        00EE              D  PRCSWL  DS     2                  ; CSWL vector contents for printer
                          D
        00F0              D          DS     3
                          D
        00F3              D  STLTYP  DS     1                  ; status line type (time vs. score)
                          D
        00F4                         DEND

                                     PAGE

                          ; define offsets into game header

        0000'          D          DSECT
                       D          ORG     0
                       D
        0000           D  HDRIRL  DS      1                 ; required interpreter release (should be 3)
        0001           D  HDRTYP  DS      1                 ; game type flags (score/time, etc.)
        0002           D  HDRREL  DS      2                 ; game release
        0004           D  HDRFRZ  DS      2                 ; log. addr. of end of frozen memory
        0006           D  HDRSTR  DS      2                 ; log. addr. of start of code
        0008           D  HDRVCB  DS      2                 ; log. addr. of vocab. table
        000A           D  HDRTHG  DS      2                 ; log. addr. of thing table
        000C           D  HDRGBV  DS      2                 ; log. addr. of global variables
        000E           D  HDRIMP  DS      2                 ; log. addr. of end of impure storage
        0010           D  HDRFLG  DS      2                 ; flags (script, etc.)
        0012           D  HDRSER  DS      6                 ; game serial no. (release date)
        0018           D  HDRSBW  DS      2                 ; log. addr. of subword table
        001A           D  HDRCKA  DS      2                 ; half of last log. addr. to checksum
        001C           D  HDRCKV  DS      2                 ; expected checksum value
                       D
                       D
                       D  ; define thing table offsets
                       D
                       D          ORG     0
                       D
        0000           D  THGATT  DS      4                 ; attribute bits
        0004           D  THGPAR  DS      1                 ; parent thing number
        0005           D  THGSIB  DS      1                 ; sibling thing number
        0006           D  THGCHD  DS      1                 ; child thing number
        0007           D  THGPRP  DS      2                 ; property list pointer
                       D
        0009                      DEND

                        C         INCLUDE ZIPMAC
                        C         PAGE
                        C
                        C
                        C ; Some useful macros
                        C
                        C+DSTZ      MACRO     ADDR
                        C+          LDA       #$00
                        C+          STA       ADDR
                        C+          STA       ADDR+1
                        C           ENDM
                        C
                        C+DASL      MACRO     ADR1,ADR2
                        C+          IFNB      <ADR2>
                        C+          LDA       ADR1
                        C+          ASL       A
                        C+          STA       ADR2
                        C+          LDA       ADR1+1
                        C+          ROL       A
                        C+          STA       ADR2+1
                        C+          ELSE
                        C+          ASL       ADR1
                        C+          ROL       ADR1+1
                        C+          ENDIF
                        C           ENDM
                        C
                        C+DLSR      MACRO     ADR1,ADR2
                        C+          IFNB      <ADR2>
                        C+          LDA       ADR1+1
                        C+          LSR       A
                        C+          STA       ADR2+1
                        C+          LDA       ADR1
                        C+          ROR       A
                        C+          STA       ADR2
                        C+          ELSE
                        C+          LSR       ADR1+1
                        C+          ROR       ADR1
                        C+          ENDIF
                        C           ENDM
                        C
                        C+DROR      MACRO     ADR1,ADR2
                        C+          IFNB      <ADR2>
                        C+          LDA       ADR1+1
                        C+          ROR       A
                        C+          STA       ADR2+1
                        C+          LDA       ADR1
                        C+          ROR       A
                        C+          STA       ADR2
                        C+          ELSE
                        C+          ROR       ADR1+1
                        C+          ROR       ADR1
                        C+          ENDIF
                        C           ENDM
                        C
                        C+DROL      MACRO     ADR1,ADR2
                        C+          IFNB      <ADR2>
                        C+          LDA       ADR1
```

```
C+          ROL       A
C+          STA       ADR2
C+          LDA       ADR1+1
C+          ROL       A
C+          STA       ADR2+1
C+          ELSE
C+          ROL       ADR1
C+          ROL       ADR1+1
C+          ENDIF
C           ENDM
C
C+DOR       MACRO     ADR1,ADR2,ADR3
C+          LDA       ADR1+1
C+          ORA       ADR2+1
C+          STA       ADR3+1
C+          LDA       ADR1
C+          ORA       ADR2
C+          STA       ADR3
C           ENDM
C
C+DAND      MACRO     ADR1,ADR2,ADR3
C+          LDA       ADR1+1
C+          AND       ADR2+1
C+          STA       ADR3+1
C+          LDA       ADR1
C+          AND       ADR2
C+          STA       ADR3
C           ENDM
C
C+D1COMP    MACRO     ADR1,ADR2
C+          LDA       ADR1
C+          EOR       #$FF
C+          STA       ADR2
C+          LDA       ADR1+1
C+          EOR       #$FF
C+          STA       ADR2+1
C           ENDM
C
C+DADC      MACRO     ADR1,ADR2,ADR3
C+          LDA       ADR1
C+          ADC       ADR2
C+          IRP       ADR,<ADR3>
C+          STA       ADR
C+          ENDM
C+          LDA       ADR1+1
C+          ADC       ADR2+1
C+          IRP       ADR,<ADR3>
C+          STA       ADR+1
C+          ENDM
C           ENDM
C
C+DSBC      MACRO     ADR1,ADR2,ADR3
C+          LDA       ADR1
C+          SBC       ADR2
C+          IRP       ADR,<ADR3>
C+          STA       ADR
```

```
C+          ENDM
C+          LDA     ADR1+1
C+          SBC     ADR2+1
C+          IRP     ADR,<ADR3>
C+          STA     ADR+1
C+          ENDM
C           ENDM
C
C+DADD      MACRO   ADR1,ADR2,ADR3
C+          CLC
C+          DADC    <ADR1>,<ADR2>,<ADR3>
C           ENDM
C
C+DSUB      MACRO   ADR1,ADR2,ADR3
C+          SEC
C+          DSBC    <ADR1>,<ADR2>,<ADR3>
C           ENDM
C
C+ADD       MACRO   ADR1,ADR2,ADR3
C+          IFNB    <ADR1>
C+          LDA     ADR1
C+          ENDIF
C+          CLC
C+          ADC     ADR2
C+          IFNB    <ADR3>
C+          IRP     ADR,<ADR3>
C+          STA     ADR
C+          ENDM
C+          ENDIF
C           ENDM
C
C+SUB       MACRO   ADR1,ADR2,ADR3
C+          IFNB    <ADR1>
C+          LDA     ADR1
C+          ENDIF
C+          SEC
C+          SBC     ADR2
C+          IFNB    <ADR3>
C+          IRP     ADR,<ADR3>
C+          STA     ADR
C+          ENDM
C+          ENDIF
C           ENDM
C
C+DADDB1    MACRO   ADDR,BYTE
C+          LOCAL   LABEL
C+          CLC
C+          LDA     ADDR
C+          ADC     BYTE
C+          STA     ADDR
C+          BCC     LABEL
C+          INC     ADDR+1
C+LABEL:
C           ENDM
C
C+DSUBB1    MACRO   ADDR,BYTE
```

```
C+          LOCAL   LABEL
C+          SEC
C+          LDA     ADDR
C+          SBC     BYTE
C+          STA     ADDR
C+          BCS     LABEL
C+          DEC     ADDR+1
C+LABEL:
C           ENDM
C
C+DADDB2    MACRO   ADDR,BYTE
C+          LOCAL   LABEL
C+          IFNB    <BYTE>
C+          ADD     ADDR,BYTE,ADDR
C+          ELSE
C+          ADD     ,ADDR,ADDR
C+          ENDIF
C+          BCC     LABEL
C+          INC     ADDR+1
C+LABEL:
C           ENDM
C
C+DSUBB2    MACRO   ADDR,BYTE
C+          LOCAL   LABEL
C+          IFNB    <BYTE>
C+          SUB     ADDR,BYTE,ADDR
C+          ELSE
C+          SUB     ,ADDR,ADDR
C+          ENDIF
C+          BCS     LABEL
C+          DEC     ADDR+1
C+LABEL:
C           ENDM
C
C+DINC      MACRO   ADDR
C+          LOCAL   LABEL
C+          INC     ADDR
C+          BNE     LABEL
C+          INC     ADDR+1
C+LABEL:
C           ENDM
C
C+DDEC      MACRO   ADDR
C+          DSUBB2  ADDR,<#$01>,ADDR
C           ENDM
C
C+DDEC2     MACRO   ADDR
C+          DSUBB2  ADDR,<#$02>,ADDR
C           ENDM
C
C+DMOV      MACRO   ADR1,ADR2
C+          LDA     ADR1
C+          IRP     ADR,<ADR2>
C+          STA     ADR
C+          ENDM
C+          LDA     ADR1+1
```

```
C+              IRP      ADR,<ADR2>
C+              STA      ADR+1
C+              ENDM
C               ENDM
C
C+DMOVI         MACRO    DATA,ADR2
C+              LDA      #<(DATA)
C+              IRP      ADR,<ADR2>
C+              STA      ADR
C+              ENDM
C+              LDA      #>(DATA)
C+              IRP      ADR,<ADR2>
C+              STA      ADR+1
C+              ENDM
C               ENDM
C
C+DMOVI2        MACRO    DATA,ADR2
C+              LDA      #>(DATA)
C+              IRP      ADR,<ADR2>
C+              STA      ADR+1
C+              ENDM
C+              LDA      #<(DATA)
C+              IRP      ADR,<ADR2>
C+              STA      ADR
C+              ENDM
C               ENDM
C
C+PUL           MACRO    ADR1
C+              IRP      ADR,<ADR1>
C+              PLA
C+              STA      ADR
C+              ENDM
C               ENDM
C
C+PSH           MACRO    ADR1
C+              IRP      ADR,<ADR1>
C+              LDA      ADR
C+              PHA
C+              ENDM
C               ENDM
C
C+DPUL          MACRO    ADR
C+              PUL      ADR+1
C+              PUL      ADR
C               ENDM
C
C+DPUL2         MACRO    ADR
C+              PUL      ADR
C+              PUL      ADR+1
C               ENDM
C
C+DPSH          MACRO    ADR
C+              PSH      ADR
C+              PSH      ADR+1
C               ENDM
C
```

```
C+MOV     MACRO    ADR1,ADR2
C+        LDA      ADR1
C+        IRP      ADR,<ADR2>
C+        STA      ADR
C+        ENDM
C         ENDM
C
C+INCA    MACRO
C+        ADD      ,<#$01>
C         ENDM
C
C+DECA    MACRO
C+        SUB      ,<#$01>
C         ENDM
C
C+TSTA    MACRO
C+        ORA      #$00
C         ENDM
C
C+STR     MACRO    TEXT
C+        DB       TEXT
C         ENDM
C
C+JEQ     MACRO    ADR
C+        LOCAL    LABEL
C+        BNE      LABEL
C+        JMP      ADR
C+LABEL:
C         ENDM
C
C+JNE     MACRO    ADR
C+        LOCAL    LABEL
C+        BEQ      LABEL
C+        JMP      ADR
C+LABEL:
C         ENDM
C
C+JCC     MACRO    ADR
C+        LOCAL    LABEL
C+        BCS      LABEL
C+        JMP      ADR
C+LABEL:
C         ENDM
C
C+JCS     MACRO    ADR
C+        LOCAL    LABEL
C+        BCC      LABEL
C+        JMP      ADR
C+LABEL:
C         ENDM
C
C+JLT     MACRO    ADR
C+        LOCAL    LABEL
C+        BGE      LABEL
C+        JMP      ADR
C+LABEL:
```

```
C       ENDM
C
C+JGE    MACRO   ADR
C+      LOCAL   LABEL
C+      BLT     LABEL
C+      JMP     ADR
C+LABEL:
C       ENDM
C
C+JGT    MACRO   ADR
C+      LOCAL   LABEL
C+      BLT     LABEL
C+      BCC     LABEL
C+      JMP     ADR
C+LABEL:
C       ENDM
C
C+JPL    MACRO   ADR
C+      LOCAL   LABEL
C+      BMI     LABEL
C+      JMP     ADR
C+LABEL:
C       ENDM
C
C+JMI    MACRO   ADR
C+      LOCAL   LABEL
C+      BPL     LABEL
C+      JMP     ADR
C+LABEL:
C       ENDM
C
C+JSREQ  MACRO   ADR,ADR2
C+      LOCAL   LABEL
C+      BNE     LABEL
C+      JSR     ADR
C+      IFNB    <ADR2>
C+      JMP     ADR2
C+      ENDIF
C+LABEL:
C       ENDM
C
C+JSRNE  MACRO   ADR,ADR2
C+      LOCAL   LABEL
C+      BEQ     LABEL
C+      JSR     ADR
C+      IFNB    <ADR2>
C+      JMP     ADR2
C+      ENDIF
C+LABEL:
C       ENDM
C
C+JSRCC  MACRO   ADR,ADR2
C+      LOCAL   LABEL
C+      BCS     LABEL
C+      JSR     ADR
C+      IFNB    <ADR2>
```

```
C+              JMP     ADR2
C+              ENDIF
C+LABEL:
C               ENDM
C
C+JSRCS         MACRO   ADR,ADR2
C+              LOCAL   LABEL
C+              BCC     LABEL
C+              JSR     ADR
C+              IFNB    <ADR2>
C+              JMP     ADR2
C+              ENDIF
C+LABEL:
C               ENDM
C
C+JSRLT         MACRO   ADR,ADR2
C+              LOCAL   LABEL
C+              BGE     LABEL
C+              JSR     ADR
C+              IFNB    <ADR2>
C+              JMP     ADR2
C+              ENDIF
C+LABEL:
C               ENDM
C
C+JSRGE         MACRO   ADR,ADR2
C+              LOCAL   LABEL
C+              BLT     LABEL
C+              JSR     ADR
C+              IFNB    <ADR2>
C+              JMP     ADR2
C+              ENDIF
C+LABEL:
C               ENDM
C
C+JSRGT         MACRO   ADR,ADR2
C+              LOCAL   LABEL
C+              BLT     LABEL
C+              BEQ     LABEL
C+              JSR     ADR
C+              IFNB    <ADR2>
C+              JMP     ADR2
C+              ENDIF
C+LABEL:
C               ENDM
C
C+JSRPL         MACRO   ADR,ADR2
C+              LOCAL   LABEL
C+              BMI     LABEL
C+              JSR     ADR
C+              IFNB    <ADR2>
C+              JMP     ADR2
C+              ENDIF
C+LABEL:
C               ENDM
C
```

```
C+JSRMI    MACRO    ADR,ADR2
C+         LOCAL    LABEL
C+         BPL      LABEL
C+         JSR      ADR
C+         IFNB     <ADR2>
C+         JMP      ADR2
C+         ENDIF
C+LABEL:
C          ENDM
C
C+RTSEQ    MACRO    ADR
C+         LOCAL    LABEL
C+         BNE      LABEL
C+         RTS
C+LABEL:
C          ENDM
C
C+RTSNE    MACRO    ADR
C+         LOCAL    LABEL
C+         BEQ      LABEL
C+         RTS
C+LABEL:
C          ENDM
C
C+RTSCC    MACRO    ADR
C+         LOCAL    LABEL
C+         BCS      LABEL
C+         RTS
C+LABEL:
C          ENDM
C
C+RTSCS    MACRO    ADR
C+         LOCAL    LABEL
C+         BCC      LABEL
C+         RTS
C+LABEL:
C          ENDM
C
C+RTSLT    MACRO    ADR
C+         LOCAL    LABEL
C+         BGE      LABEL
C+         RTS
C+LABEL:
C          ENDM
C
C+RTSGE    MACRO    ADR
C+         LOCAL    LABEL
C+         BLT      LABEL
C+         RTS
C+LABEL:
C          ENDM
C
C+RTSGT    MACRO    ADR
C+         LOCAL    LABEL
C+         BLT      LABEL
C+         BEQ      LABEL
```

```
C+          RTS
C+LABEL:
C           ENDM
C
C+RTSPL     MACRO     ADR
C+          LOCAL     LABEL
C+          BMI       LABEL
C+          RTS
C+LABEL:
C           ENDM
C
C+RTSMI     MACRO     ADR
C+          LOCAL     LABEL
C+          BPL       LABEL
C+          RTS
C+LABEL:
C           ENDM
C
C+DTST      MACRO     ADDR
C+          LDA       ADDR+1
C+          ORA       ADDR
C           ENDM
C
C+DTSTBE    MACRO     ADR1,ADR2
C+          DTST      ADR1
C+          BEQ       ADR2
C           ENDM
C
C+DTSTBN    MACRO     ADR1,ADR2
C+          DTST      ADR1
C+          BNE       ADR2
C           ENDM
C
C+DTSTJE    MACRO     ADR1,ADR2
C+          DTST      ADR1
C+          JEQ       ADR2
C           ENDM
C
C+DTSTJN    MACRO     ADR1,ADR2
C+          DTST      ADR1
C+          JNE       ADR2
C           ENDM
C
C+DTSTRE    MACRO     ADR1
C+          DTST      ADR1
C+          RTSEQ
C           ENDM
C
C+DTSTRN    MACRO     ADR1
C+          DTST      ADR1
C+          RTSNE
C           ENDM
C
C+DTST2     MACRO     ADDR
C+          LDA       ADDR
C+          ORA       ADDR+1
```

```
C          ENDM
C
C+DTS2BE    MACRO    ADR1,ADR2
C+          DTST2    ADR1
C+          BEQ      ADR2
C           ENDM
C
C+DTS2BN    MACRO    ADR1,ADR2
C+          DTST2    ADR1
C+          BNE      ADR2
C           ENDM
C
C+DTS2JE    MACRO    ADR1,ADR2
C+          DTST2    ADR1
C+          JEQ      ADR2
C           ENDM
C
C+DTS2JN    MACRO    ADR1,ADR2
C+          DTST2    ADR1
C+          JNE      ADR2
C           ENDM
C
C+DTS2RE    MACRO    ADR1
C+          DTST2    ADR1
C+          RTSEQ
C           ENDM
C
C+DTS2RN    MACRO    ADR1
C+          DTST2    ADR1
C+          RTSNE
C           ENDM
C
C+DXBNE     MACRO    ADR
C+          DEX
C+          BNE      ADR
C           ENDM
C
C+DYBNE     MACRO    ADR
C+          DEY
C+          BNE      ADR
C           ENDM
C
C+DXBEQ     MACRO    ADR
C+          DEX
C+          BEQ      ADR
C           ENDM
C
C+DYBEQ     MACRO    ADR
C+          DEY
C+          BEQ      ADR
C           ENDM
C
C+DXBPL     MACRO    ADR
C+          DEX
C+          BPL      ADR
C           ENDM
```

```
C
C+DYBPL   MACRO   ADR
C+        DEY
C+        BPL     ADR
C         ENDM
C
C+DXBMI   MACRO   ADR
C+        DEX
C+        BMI     ADR
C         ENDM
C
C+DYBMI   MACRO   ADR
C+        DEY
C+        BMI     ADR
C         ENDM
C
C+IXBNE   MACRO   ADR
C+        INX
C+        BNE     ADR
C         ENDM
C
C+IYBNE   MACRO   ADR
C+        INY
C+        BNE     ADR
C         ENDM
C
C+DECBE   MACRO   ADR1,ADR2
C+        DEC     ADR1
C+        BEQ     ADR2
C         ENDM
C
C+DECBN   MACRO   ADR1,ADR2
C+        DEC     ADR1
C+        BNE     ADR2
C         ENDM
C
C+DECJE   MACRO   ADR1,ADR2
C+        DEC     ADR1
C+        JEQ     ADR2
C         ENDM
C
C+DECJN   MACRO   ADR1,ADR2
C+        DEC     ADR1
C+        JNE     ADR2
C         ENDM
C
C+DECABE  MACRO   ADR1
C+        DECA
C+        BEQ     ADR1
C         ENDM
C
C+DECABN  MACRO   ADR1
C+        DECA
C+        BNE     ADR1
C         ENDM
C
```

```
C+DECABP   MACRO    ADR1
C+         DECA
C+         BPL      ADR1
C          ENDM
C
C+DECABM   MACRO    ADR1
C+         DECA
C+         BMI      ADR1
C          ENDM
C
C+TSTABE   MACRO    ADR1
C+         TSTA
C+         BEQ      ADR1
C          ENDM
C
C+TSTABN   MACRO    ADR1
C+         TSTA
C+         BNE      ADR1
C          ENDM
C
C+TSTABP   MACRO    ADR1
C+         TSTA
C+         BPL      ADR1
C          ENDM
C
C+TSTABM   MACRO    ADR1
C+         TSTA
C+         BMI      ADR1
C          ENDM
C
C+TSTAJE   MACRO    ADR1
C+         TSTA
C+         JEQ      ADR1
C          ENDM
C
C+TSTARP   MACRO
C+         TSTA
C+         RTSPL
C          ENDM
C
C+CMPBE    MACRO    ADR1,ADR2
C+         CMP      ADR1
C+         BEQ      ADR2
C          ENDM
C
C+CMPBN    MACRO    ADR1,ADR2
C+         CMP      ADR1
C+         BNE      ADR2
C          ENDM
C
C+CMPBL    MACRO    ADR1,ADR2
C+         CMP      ADR1
C+         BLT      ADR2
C          ENDM
C
C+CMPBG    MACRO    ADR1,ADR2
```

```
C+              CMP     ADR1
C+              BGE     ADR2
C               ENDM
C
C+CMPBM         MACRO   ADR1,ADR2
C+              CMP     ADR1
C+              BMI     ADR2
C               ENDM
C
C+CMPBP         MACRO   ADR1,ADR2
C+              CMP     ADR1
C+              BPL     ADR2
C               ENDM
C
C+CMPJE         MACRO   ADR1,ADR2
C+              CMP     ADR1
C+              JEQ     ADR2
C               ENDM
C
C+CMPJL         MACRO   ADR1,ADR2
C+              CMP     ADR1
C+              JLT     ADR2
C               ENDM
C
C+CMPJSE        MACRO   ADR1,ADR2
C+              CMP     ADR1
C+              JSREQ   ADR2
C               ENDM
C
C+CMPJSN        MACRO   ADR1,ADR2
C+              CMP     ADR1
C+              JSRNE   ADR2
C               ENDM
C
C+CMPJSG        MACRO   ADR1,ADR2
C+              CMP     ADR1
C+              JSRGE   ADR2
C               ENDM
C
C+CMPRE         MACRO   ADR1
C+              CMP     ADR1
C+              RTSEQ
C               ENDM
C
C+CPXBE         MACRO   ADR1,ADR2
C+              CPX     ADR1
C+              BEQ     ADR2
C               ENDM
C
C+CPXBG         MACRO   ADR1,ADR2
C+              CPX     ADR1
C+              BGE     ADR2
C               ENDM
C
C+CPXRGT        MACRO   ADR1
C+              CPX     ADR1
```

```
C+              RTSGT
C               ENDM
C
C+CPYBN         MACRO       ADR1,ADR2
C+              CPY         ADR1
C+              BNE         ADR2
C               ENDM

                PAGE
```

```

                            ; start of interpreter

    0000'                            ASEG
                                     ORG      LDORG                   ; load at one address
                                     .PHASE   MAINOR                  ; but assemble for another

    0800     D8             START:   CLD                              ; very important

    0801     A9 00                   LDA      #$00                    ; clear our section of zero page
    0803     A2 80                   LDX      #$80
    0805     95 00          L0805:   STA      $00,X
                           +         IXBNE    L0805

    080A     A2 FF                   LDX      #$FF                    ; init hardware stack
    080C     9A                      TXS

    080D     20 1AF7                 JSR      INITSC                  ; init and clear screen window

                           +         MOV      <#$00>,<PRGUPD,AUXUPD>  ; indicate no pages loaded

                           +         MOV      <#$01>,STKCNT           ; init software stack
                           +         DMOVI    STCKLC,STKPNT

                           +         MOV      <#$FF>,LD9

                           +         DMOVI    VMT1LC,VMTAB1           ; init virtual memory table pointers
                           +         DMOVI    VMT2LC,VMTAB2
                           +         DMOVI    VMT3LC,VMTAB3
                           +         DMOVI    VMT4LC,VMTAB4

    0846     A0 00                   LDY      #$00                    ; init virtual memory tables
    0848     A2 80                   LDX      #$80
    084A                   +L084A:   MOV      <#$FF>,<<(VMTAB1),Y>,<(VMTAB2),Y>>
    0850     98                      TYA
                           +         ADD      ,<#$01>,<<(VMTAB3),Y>>
    0856     98                      TYA
                           +         SUB      ,<#$01>,<<(VMTAB4),Y>>
    085C     C8                      INY
                           +         DXBNE    L084A
    0860     88                      DEY
                           +         MOV      <#$FF>,<<(VMTAB3),Y>>

                           +         MOV      <#$00>,MRUPAG
                           +         MOV      <#$7F>,LRUPAG

                           +         DMOVI    FIRFLC,FRZMEM           ; init memory pointers

                           +         DMOV     FRZMEM,ACC              ; read log page 0 to first frozen page
                           +         DMOVI    $0000,ACB
    0885     20 1E0D                 JSR      DRDBKF

    0888     A0 05                   LDY      #HDRFRZ+1               ; setup frozen storage page count
                           +         MOV      <#$FF>,<<(FRZMEM),Y>>   ; bump up to page boundary-1
    088E     88                      DEY
```

```
                        +         MOV     <(FRZMEM),Y>,FRZPGS
0893    E6 BC                     INC     FRZPGS

0895    A9 00                     LDA     #$00                  ; read in rest of frozen memory
0897                    +L0897:   ADD     ,<#$01>
089A    AA                        TAX
089B    65 BB                     ADC     FRZMEM+1
089D    85 E7                     STA     ACC+1
                        +         MOV     FRZMEM,ACC
08A3    8A                        TXA
                        +         CMPBE   FRZPGS,L08B6
08A8    48                        PHA
08A9    85 E4                     STA     ACB
                        +         MOV     <#$00>,ACB+1
08AF    20 1E0D                   JSR     DRDBKF
08B2    68                        PLA
08B3    4C 0897                   JMP     L0897

08B6    A0 01            L08B6:   LDY     #HDRTYP               ; setup for proper type of status line
08B8    B1 BA                     LDA     (FRZMEM),Y
08BA    29 02                     AND     #$02
08BC    85 F3                     STA     STLTYP

08BE    A0 07                     LDY     #HDRSTR+1             ; init PC
                        +         MOV     <(FRZMEM),Y>,PRGIDX
08C4    88                        DEY
                        +         MOV     <(FRZMEM),Y>,PRGLPG
                        +         MOV     <#$00>,PRGLPG+1

08CD    A0 0D                     LDY     #HDRGBV+1             ; init global variable pointer
                        +         MOV     <(FRZMEM),Y>,GLBVAR
08D3    88                        DEY
                        +         ADD     <(FRZMEM),Y>,FRZMEM+1,GLBVAR+1

08DB    A0 19                     LDY     #HDRSBW+1             ; init sub-word table pointer
                        +         MOV     <(FRZMEM),Y>,SBWDPT
08E1    88                        DEY
                        +         ADD     <(FRZMEM),Y>,FRZMEM+1,SBWDPT+1

                        +         MOV     <#$00>,SWPMEM         ; swpmem := frzmem + 256 * frzpgs
                        +         ADD     FRZPGS,FRZMEM+1,SWPMEM+1

08F4    20 1B1E                   JSR     FNDMEM                ; determine number of pages of memory
                        +         SUB     ,SWPMEM+1             ; swppgs := (maxmem - swpmem) / 256
08FA    90 0E                     BCC     L090A                 ; if swppgs < 0 then fatal error
08FC    A8                        TAY
08FD    C8                        INY
08FE    84 BD                     STY     SWPPGS
0900    A8                        TAY
0901    84 BF                     STY     LRUPAG
                        +         MOV     <#$FF>,<<(VMTAB3),Y>>

0907    4C 098F                   JMP     MNLOOP                ; start the game!

090A    20 21D1          L090A:   JSR     FATAL

```

```
1131                                PAGE
```

```

                        ; class C instructions (implicit or no operand)

090D    0C18            OPTAB1: DW      OPRTNT                  ; return with TRUE
090F    0C23                    DW      OPRTNF                  ; return with FALSE
0911    0C28                    DW      OPPSI                   ; print string immediate
0913    0C54                    DW      OPPSIC                  ; print string immediate, CRLF, return true
0915    0C53                    DW      OPNULL                  ; no-op
0917    204B                    DW      OPSVGM                  ; save game status to disk
0919    20EB                    DW      OPRSGM                  ; restore game status from disk
091B    0800                    DW      START                   ; restart game
091D    0C64                    DW      OPRTNV                  ; return with value
091F    1720                    DW      PULLWD                  ; drop a word from the stack
0921    21EA                    DW      OPENDS                  ; end the game
0923    0C72                    DW      OPCRLF                  ; print CRLF
0925    1C8A                    DW      OPPRST                  ; print status line
0927    0C7C                    DW      OPCKSM                  ; checksum the program
000E                    OPMAX1  EQU     (*-OPTAB1)/2


                        ; class B instructions (single operand)

0929    0CDD            OPTAB2: DW      OPTSTZ                  ; compare ARG1=0 (ARG1<>0)
092B    0CE9                    DW      OPGTSB                  ; get thing's sibling
092D    0CF3                    DW      OPGTCH                  ; get thing's child
092F    0D0E                    DW      OPGTPR                  ; get thing's parent
0931    0D20                    DW      OPGTPL                  ; get length of property (given addr)
0933    0D43                    DW      OPINC                   ; increment variable
0935    0D60                    DW      OPDEC                   ; decrement variable
0937    0D73                    DW      OPPSB                   ; print string at byte address
0939    21D1                    DW      FATAL
093B    0D81                    DW      OPDSTT                  ; destroy thing
093D    0DE2                    DW      OPPRTN                  ; print thing name
093F    0E06                    DW      OPRTN                   ; return
0941    0E7C                    DW      OPJUMP                  ; unconditional jump
0943    0E92                    DW      OPPSW                   ; print string at word address
0945    0EA0                    DW      OPMOVE                  ; move var ARG1 to var
0947    0EA8                    DW      OPNOT                   ; 1's complement
0010                    OPMAX2  EQU     (*-OPTAB2)/2

                                PAGE

                                ; class A instructions (variable number of operands, may use short form
                                ; opcode)

        0949    21D1            OPTAB3: DW      FATAL
        094B    116B                    DW      OPMTCH                  ; match ARG1 against ARG2, ARG3, or ARG4
        094D    0EB7                    DW      LOEB7                   ; ??? compare ARG1<=ARG2 (ARG1>ARG2)
        094F    0ECF                    DW      LOECF                   ; ??? compare ARG1>=ARG2 (ARG1<ARG2)
        0951    0EE7                    DW      OPDECB                  ; decrement variable and branch
        0953    0EF5                    DW      OPINCB                  ; increment variable and branch
        0955    0F13                    DW      OPTINT                  ; is thing ARG1 in thing ARG2
        0957    0F23                    DW      LOF23
        0959    0F3B                    DW      OPOR                    ; logical OR
        095B    0F4A                    DW      OPAND                   ; logical AND
        095D    0F59                    DW      OPTSTA                  ; test thing attribute
        095F    0F6D                    DW      OPSETA                  ; set thing attribute
        0961    0F80                    DW      OPCLRA                  ; clear thing attribute
        0963    0F97                    DW      LOF97                   ; move ARG2 into var ARG1
        0965    0FA4                    DW      OPMOVT                  ; move thing ARG1 into thing ARG2
        0967    0FD2                    DW      OPGTWD                  ; get a word
        0969    0FEC                    DW      OPGTBY                  ; store a word
        096B    1008                    DW      OPGTP                   ; get thing property
        096D    1069                    DW      OPGTPA                  ; get address of property
        096F    109E                    DW      OPGTNP                  ; get next property
        0971    10C3                    DW      OPADD                   ; add
        0973    10D3                    DW      OPSUB                   ; subtract
        0975    10E3                    DW      OPMUL                   ; multiply
        0977    1118                    DW      OPDIV                   ; divide
        0979    114A                    DW      OPRMD                   ; remainder
        0019                    OPMAX3  EQU     (*-OPTAB3)/2


                                ; class D instructions (variable number of operands)

        097B    11A3            OPTAB4: DW      OPCALL                  ; call procedure
        097D    125F                    DW      OPPTWD                  ; store a word
        097F    1288                    DW      OPPTBY                  ; store a byte
        0981    12A9                    DW      OPPTP                   ; store into thing property
        0983    12DC                    DW      OPGTLN                  ; get a line of input
        0985    14E5                    DW      OPPRCH                  ; print a character
        0987    14EA                    DW      OPPRNM                  ; print number
        0989    1536                    DW      OPRNDM                  ; generate random number
        098B    1555                    DW      OPPUSH                  ; push ARG1 to stack
        098D    1560                    DW      OPPULL                  ; pull var from stack
        000A                    OPMAX4  EQU     (*-OPTAB4)/2

                                        PAGE
```

```


 098F                +MNLOOP: MOV     <#$00>,ARGCNT           ; default no arguments

 0993    20 173E              JSR     FTPRBA                  ; get opcode
 0996    85 80                STA     OPCODE

                     +        CMPJL   #$80,OPCGPA             ; is it class A ($00-$7F)?
                     +        CMPJL   #$B0,OPCGPB             ; how about class B ($80-$AF)?
                     +        CMPBL   #$C0,OPCGPC             ; perhaps class C ($B0-$BF)?
                      ;       JMP     OPCGPD                  ; nope, it's class D ($C0-$FF).


                      ; process opcode group D ($C0-$FF)

 09AA    20 173E      OPCGPD: JSR     FTPRBA                  ; get operand specification byte

 09AD    A2 00                LDX     #$00                    ; init operand count

 09AF    48           L09AF:  PHA                             ; save the operand specification byte
 09B0    A8                   TAY                             ;   in Y and on stack

 09B1    8A                   TXA                             ; save operand count on stack
 09B2    48                   PHA

 09B3    98                   TYA                             ; get back operand specification byte
 09B4    29 C0                AND     #$C0                    ; look at top two bits

                     +        JSREQ   FTPRWD,L09D7            ; if they're 00, operand is word immed.
                     +        CMPJSE  #$80,<GTVARP,L09D7>     ; 10? variable
                     +        CMPJSE  #$40,<FTPRBY,L09D7>     ; 01? byte immediate

 09D2    68                   PLA                             ; must be 11, no more operands
 09D3    68                   PLA                             ; pull operand spec byte and count
 09D4    4C 09ED              JMP     L09ED                   ; and finish up

 09D7    68           L09D7:  PLA                             ; get operand count back
 09D8    AA                   TAX                             ; to use as index

                     +        MOV     ACC,<<ARG1,X>>          ; store operand in proper ARG location
                     +        MOV     ACC+1,<<ARG1+1,X>>

 09E1    E8                   INX                             ; increment ARG pointer
 09E2    E8                   INX
 09E3    E6 81                INC     ARGCNT                  ; and count

 09E5    68                   PLA                             ; pull arg spec byte
 09E6    38                   SEC                             ; shift top two bits off left, while
 09E7    2A                   ROL     A                       ; shifting 11 in from right (to
 09E8    38                   SEC                             ; indicate no more operands)
 09E9    2A                   ROL     A

 09EA    4C 09AF              JMP     L09AF                   ; try for another

 09ED                +L09ED:  DMOVI   OPTAB4,ACC              ; assume class D
```

```
09F5   A5 80                  LDA    OPCODE              ; but if it's $C0-$DF then it's class A
                    +         CMPJL  #$E0,L0A98

09FE   E9 E0                  SBC    #$E0                ; adjust to $00..$1F
                    +         CMPBG  #OPMAX4,L0A2B       ; make sure it's not illegal

0A04   0A             GODOIT: ASL    A                   ; get address from table (base in ACC)
0A05   A8                     TAY                        ;   word indexed by A and execute
                    +         MOV    <(ACC),Y>,DSPTCH+1
0A0B   C8                     INY
                    +         MOV    <(ACC),Y>,DSPTCH+2
0A11   20 0A11        DSPTCH: JSR    DSPTCH
0A14   4C 098F                JMP    MNLOOP


                      ; process opcode group C ($B0-$BF)

0A17                +OPCGPC:  SUB    ,<#$B0>             ; adjust to $00..$0F
                    +         CMPBG  #OPMAX1,L0A2B       ; make sure it's not illegal
0A1E   48                     PHA                        ; save it temp.
                    +         DMOVI  OPTAB1,ACC          ; get base address of proper table
0A27   68                     PLA
0A28   4C 0A04                JMP    GODOIT

0A2B   20 21D1        L0A2B:  JSR    FATAL               ; oops!  illegal opcode


                      ; process opcode group B ($80-$AF)

0A2E   29 30          OPCGPB: AND    #$30                ; mask off operand type bits

                    +         JSREQ  FTPRWD,L0A45        ; 00?  then it's word immediate
                    +         CMPJSE #$10,<FTPRBY,L0A45> ; 01?  byte immediate
0A42   20 0AE8                JSR    GTVARP              ; must be 10, variable

0A45                +L0A45:   MOV    <#$01>,ARGCNT       ; one argument
                    +         DMOV   ACC,ARG1

0A51   A5 80                  LDA    OPCODE              ; adjust opcode to $00..$0F
0A53   29 0F                  AND    #$0F
                    +         CMPBG  #OPMAX2,L0A2B       ; make sure it's not illegal
0A59   48                     PHA                        ; save temp.
                    +         DMOVI  OPTAB2,ACC          ; get appropriate table base addr
0A62   68                     PLA
0A63   4C 0A04                JMP    GODOIT              ; and go do it!


                      ; process opcode group A ($00-$7F)

0A66   29 40          OPCGPA: AND    #$40                ; get type bit for ARG1
                    +         JSREQ  FTPRBY,L0A73        ; 0:  byte immediate
0A70   20 0AE8                JSR    GTVARP              ; 1:  variable/stack
0A73                +L0A73:   DMOV   ACC,ARG1            ; save it

0A7B   A5 80                  LDA    OPCODE              ; get type bit for ARG2
0A7D   29 20                  AND    #$20
```

```
                         +         JSREQ   FTPRBY,LOA8A          ; 0:  byte immediate
 0A87     20 0AE8                  JSR     GTVARP                ; 1:  variable/stack
 0A8A                    +LOA8A:   DMOV    ACC,ARG2              ; save it

                         +         MOV     <#$02>,ARGCNT         ; indicate two operands

 0A96     A5 80                    LDA     OPCODE                ; get opcode back
 0A98     29 1F           LOA98:   AND     #$1F                  ; adjust to $00..$1F
                         +         CMPBG   #OPMAX3,LOA2B         ; make sure it's not illegal
 0A9E     48                       PHA                           ; save temp.
                         +         DMOVI   OPTAB3,ACC            ; get base addr of appropriate table
 0AA7     68                       PLA
 0AA8     4C 0A04                  JMP     GODOIT                ; and go do it!

                                   PAGE
```

```

                          ; fetch byte immediate into ACC

        0AAB    20 173E    FTPRBY: JSR     FTPRBA              ; get a byte from program into A
        0AAE    85 E6              STA     ACC                 ; zero-fill to 16 bits in ACC
                          +        MOV     <#$00>,ACC+1
        0AB4    60                 RTS                         ; return


                          ; fetch word immediate into ACC

        0AB5    20 173E    FTPRWD: JSR     FTPRBA              ; get high byte from program into A
        0AB8    48                 PHA                         ; save it temp.
        0AB9    20 173E            JSR     FTPRBA              ; get low byte from program into A
        0ABC    85 E6              STA     ACC                 ; store low byte
                          +        PUL     ACC+1               ; store high byte
        0AC1    60                 RTS                         ; return


        0AC2              +GTVRA1: TSTABE  LOAD0               ; fetch ACC from var in A, keep if stack
        0AC6    4C 0AEF            JMP     GTVARA


        0AC9              +PTVRA1: TSTABE  LOAD6               ; store ACC into var in A, replace if stack
        0ACD    4C 0B46            JMP     PTVARA

        0AD0    20 1720     LOAD0: JSR     PULLWD              ; read stack non-destructive
        0AD3    4C 16F4            JMP     PUSHWD

        0AD6              +LOAD6:  DPSH    ACC                 ; replace TOS w/ ACC
        0ADC    20 1720            JSR     PULLWD
                          +        DPUL    ACC
        0AE5    4C 16F4            JMP     PUSHWD

                                   PAGE
```

```

0AE8     20 173E      GTVARP: JSR     FTPRBA              ; fetch ACC from var ind. by program
                     +        TSTABE  L0B26
0AEF                 +GTVARA: CMPBG   <#$10>,L0B02        ; fetch ACC from var in A
                     +        SUB     ,<#$01>
0AF6     0A                   ASL     A
0AF7     AA                   TAX
                     +        MOV     <LOCVAR,X>,ACC+1
0AFC     E8                   INX
                     +        MOV     <LOCVAR,X>,ACC
0B01     60                   RTS

0B02                 +L0B02:  SUB     ,<#$10>
0B05     0A                   ASL     A
0B06     85 E4                STA     ACB
0B08     A9 00                LDA     #$00
0B0A     2A                   ROL     A
0B0B     85 E5                STA     ACB+1
                     +        DADD    GLBVAR,ACB,ACB
0B1A     A0 00                LDY     #$00
                     +        MOV     <(ACB),Y>,ACC+1
0B20     C8                   INY
                     +        MOV     <(ACB),Y>,ACC
0B25     60                   RTS

0B26     20 1720      L0B26:  JSR     PULLWD
0B29     60                   RTS

                              PAGE
```

```

0B2A    A9 00          PTVRPZ: LDA     #$00                        ; store 0 in var. ind. by program
0B2C    85 E6          PTVRPA: STA     ACC                         ; store byte in A in var. ind. by prog.
                      +        MOV     <#$00>,ACC+1
0B32    4C 0B35        PTVRP1: JMP     PTVARP                      ; unnecessary!!!

0B35                  +PTVARP: DPSH    ACC                         ; store ACC in var. ind. by program
0B3B    20 173E                JSR     FTPRBA
0B3E    AA                     TAX
                      +        DPUL    ACC
0B45    8A                     TXA
0B46                  +PTVARA: TSTAJE  PUSHWD                      ; store ACC in var. in A
                      +        CMPBG   <#$10>,L0B60
                      +        DECA
0B54    0A                     ASL     A
0B55    AA                     TAX
                      +        MOV     ACC+1,<<LOCVAR,X>>
0B5A    E8                     INX
                      +        MOV     ACC,<<LOCVAR,X>>
0B5F    60                     RTS

0B60                  +L0B60:  SUB     ,<#$10>
0B63    0A                     ASL     A
0B64    85 E4                  STA     ACB
0B66    A9 00                  LDA     #$00
0B68    2A                     ROL
0B69    85 E5                  STA     ACB+1
                      +        DADD    GLBVAR,ACB,ACB
0B78    A0 00                  LDY     #$00
                      +        MOV     ACC+1,<<(ACB),Y>>
0B7E    C8                     INY
                      +        MOV     ACC,<<(ACB),Y>>
0B83    60                     RTS

                               PAGE
```

```

0B84    20 173E      PREDTR: JSR     FTPRBA                   ; fetch first displacement byte
                    +        TSTABM  L0B9C                    ; complement condition if necessary
0B8B    10 07                BPL     L0B94

0B8D    20 173E      PREDFL: JSR     FTPRBA                   ; fetch first displacement byte
                    +        TSTABP  L0B9C                    ; complement condition if necessary
                     ;       BMI     L0B94

0B94    29 40        L0B94:  AND     #$40                     ; branch not taken
                    +        JSREQ   FTPRBA                   ; fetch second displacement byte if
0B9B    60                   RTS                              ; necessary and discard it

0B9C    AA           L0B9C:  TAX                              ; branch take, save first disp. byte
0B9D    29 40                AND     #$40                     ; do we need a second byte?
0B9F    F0 0C                BEQ     L0BAD                    ; yes
0BA1    8A                   TXA                              ; no, extend what we have w/ zeros
0BA2    29 3F                AND     #$3F
0BA4    85 E6                STA     ACC
                    +        MOV     <#$00>,ACC+1
0BAA    4C 0BC3              JMP     L0BC3                    ; and go do it!

0BAD    8A           L0BAD:  TXA                              ; get rest of displacement
0BAE    29 3F                AND     #$3F
0BB0    48                   PHA
0BB1    20 173E              JSR     FTPRBA
0BB4    85 E6                STA     ACC
                    +        PUL     ACC+1
0BB9    29 20                AND     #$20
0BBB    F0 06                BEQ     L0BC3
0BBD    A5 E7                LDA     ACC+1
0BBF    09 C0                ORA     #$C0
0BC1    85 E7                STA     ACC+1

0BC3                +L0BC3:  DTSTBE  ACC,OPRTNF               ; if displacement = 0, return false
                    +        DDEC    ACC
                    +        DTSTBE  ACC,OPRTNT               ; if displacement = 1, return true
0BDA                +L0BDA:  DDEC    ACC

                    +        MOV     ACC+1,ACB                ; copy high byte of displacement to ACB
0BE9    0A                   ASL     A                        ; and sign extend to 17 bits
0BEA    A9 00                LDA     #$00
0BEC    2A                   ROL     A
0BED    85 E5                STA     ACB+1

                    +        ADD     PRGIDX,ACC               ; add low byte of displacement to PC
0BF4    90 06                BCC     L0BFC                    ; increment high 9 bits of displacement
                    +        DINC    ACB                      ;   if overflow
0BFC    85 8A        L0BFC:  STA     PRGIDX

                    +        DTSTBE  ACB,L0C17                ; if high 9 bits of disp. =0, all done

0C04    18                   CLC                              ; add high 9 bits of disp. to PC log page
0C05    A5 E4                LDA     ACB
0C07    65 8B                ADC     PRGLPG
```

```
0C09    85 8B                 STA     PRGLPG
0C0B    A5 E5                 LDA     ACB+1
0C0D    65 8C                 ADC     PRGLPG+1
0C0F    29 01                 AND     #$01                    ; mod 2^17
0C11    85 8C                 STA     PRGLPG+1

                    +         MOV     <#$00>,PRGUPD           ; indicate page change

0C17    60           LOC17:   RTS                             ; all done

                              PAGE
```

```

  0C18     A9 01        OPRTNT: LDA     #$01                 ; return true ($01)
  0C1A     85 82        LOC1A:  STA     ARG1                 ; return byte in A
                       +        MOV     <#$00>,ARG1+1        ; make high byte of return value $00
  0C20     4C 0E06              JMP     OPRTN                ; and do the return!

  0C23     A9 00        OPRTNF: LDA     #$00                 ; return false ($00)
  0C25     4C 0C1A              JMP     LOC1A


  0C28                 +OPPSI:  MOV     PRGIDX,AUXIDX        ; copy PC to AUX
                       +        DMOV    PRGLPG,AUXLPG
                       +        MOV     <#$00>,AUXUPD        ; indicate new log. page

  0C38     20 18B4              JSR     PRNTST               ; print the string

                       +        MOV     AUXIDX,PRGIDX        ; copy AUX back to PC
                       +        DMOV    AUXLPG,PRGLPG
                       +        MOV     AUXUPD,PRGUPD
                       +        DMOV    AUXMPT,PRGMPT

  0C53     60           OPNULL: RTS                          ; done


  0C54     20 0C28      OPPSIC: JSR     OPPSI                ; print string immediate

  0C57     A9 0D                LDA     #CRCHAR              ; print CRLF (could use JSR OPCRLF)
  0C59     20 1B3F              JSR     BFCHAR
  0C5C     A9 0A                LDA     #LFCHAR
  0C5E     20 1B3F              JSR     BFCHAR

  0C61     4C 0C18              JMP     OPRTNT               ; return true


  0C64     20 1720      OPRTNV: JSR     PULLWD               ; pull value off stack
                       +        DMOV    ACC,ARG1             ; save it for posterity
  0C6F     4C 0E06              JMP     OPRTN                ; return with it


  0C72     A9 0D        OPCRLF: LDA     #CRCHAR              ; print CRLF
  0C74     20 1B3F              JSR     BFCHAR
  0C77     A9 0A                LDA     #LFCHAR
  0C79     4C 1B3F              JMP     BFCHAR               ; implicit RTS

                                PAGE
```

```
0C7C     A0 1B        OPCKSM: LDY      #HDRCKA+1              ; get checksum end log. address (word
                      +       MOV      <(FRZMEM),Y>,ARG2      ;   index)
0C82     88                   DEY
                      +       MOV      <(FRZMEM),Y>,ARG2+1

                      +       MOV      <#$00>,<ARG3,ARG1,ARG1+1,ACC+1,ARG4>    ; initialize everything

                      +       MOV      <#ARG4>,L1807+1        ; patch VM routine to swap  in all pages

0C98     06 84                ASL      ARG2                   ; convert end address to byte index
0C9A     26 85                ROL      ARG2+1
0C9C     26 86                ROL      ARG3

                      +       MOV      <#$40>,ACC             ; start at log. address $00040
0CA2     20 17B8              JSR      SETAXB

0CA5     20 17E8      LOCA5:  JSR      FTAXBA                 ; get a byte
                      +       DADDB2   ARG1                   ;   and add it to checksum

0CB1     A5 93                LDA      AUXIDX                 ; compare AUX to end address
                      +       CMPBN    ARG2,LOCA5             ; if not done, loop
0CB7     A5 91                LDA      AUXLPG
                      +       CMPBN    ARG2+1,LOCA5
0CBD     A5 92                LDA      AUXLPG+1
                      +       CMPBN    ARG3,LOCA5

                      +       MOV      <#FRZPGS>,L1807+1      ; unpatch VM routine

0CC8     A0 1D                LDY      #HDRCKV+1              ; compare computed vs. expected checksum
0CCA     B1 BA                LDA      (FRZMEM),Y
                      +       CMPBN    ARG1,LOCDA
0CD0     88                   DEY
0CD1     B1 BA                LDA      (FRZMEM),Y
                      +       CMPJE    ARG1+1,PREDTR

0CDA     4C 0B8D      LOCDA:  JMP      PREDFL

                              PAGE
```

```

0CDD                    +OPTSTZ: DTSTJN  ARG1,PREDFL
0CE6    4C 0B84          LOCE6:  JMP     PREDTR

0CE9    A5 82            OPGTSB: LDA     ARG1                    ; get sibling of thing, predicate
0CEB    20 16A7                  JSR     SETUPT
0CEE    A0 05                    LDY     #THGSIB
0CF0    4C 0CFA                  JMP     LOCFA

0CF3    A5 82            OPGTCH: LDA     ARG1                    ; get child of thing, predicate
0CF5    20 16A7                  JSR     SETUPT
0CF8    A0 06                    LDY     #THGCHD
0CFA                    +LOCFA:  PSH     <<(ACC),Y>>
0CFD    85 E6                    STA     ACC
                        +        MOV     <#$00>,ACC+1
0D03    20 0B35                  JSR     PTVARP
0D06    68                       PLA
                        +        TSTABN  LOCE6
0D0B    4C 0B8D                  JMP     PREDFL

0D0E    A5 82            OPGTPR: LDA     ARG1                    ; get parent of thing
0D10    20 16A7                  JSR     SETUPT
0D13    A0 04                    LDY     #THGPAR
                        +        MOV     <(ACC),Y>,ACC
                        +        MOV     <#$00>,ACC+1
0D1D    4C 0B32                  JMP     PTVRP1

0D20                    +OPGTPL: DADD    ARG1,FRZMEM,ACC
                        +        DDEC    ACC
0D38    A0 00                    LDY     #$00
0D3A    20 1693                  JSR     GTPLEN
                        +        ADD     ,<#$01>
0D40    4C 0B2C                  JMP     PTVRPA


                         ; increment variable ARG1
0D43    A5 82            OPINC:  LDA     ARG1
0D45    20 0AC2                  JSR     GTVRA1
                        +        DINC    ACC
0D4E                    +LOD4E:  DPSH    ACC
0D54    A5 82                    LDA     ARG1
0D56    20 0AC9                  JSR     PTVRA1
                        +        DPUL    ACC
0D5F    60                       RTS


                         ; decrement variable ARG1

0D60    A5 82            OPDEC:  LDA     ARG1
0D62    20 0AC2                  JSR     GTVRA1
                        +        DDEC    ACC
0D70    4C 0D4E                  JMP     LOD4E


                         ; print string at byte address in ARG1
```

```

0D73                   +OPPSB:  DMOV    ARG1,ACC                ; set AUX to point to string at
0D7B     20 17B8                JSR     SETAXB                  ;   byte address
0D7E     4C 0E9D                JMP     L0E9D                   ; and print it!

                                PAGE
```

```
                             ; destroy thing ARG1 (move to location 0)

        0D81    A5 82        OPDSTT: LDA     ARG1
        0D83    20 16A7              JSR     SETUPT
        0D86    A0 04                LDY     #THGPAR
        0D88    B1 E6                LDA     (ACC),Y
                            +        RTSEQ
        0D8D    AA                   TAX
                            +        DPSH    ACC
        0D94    8A                   TXA
        0D95    20 16A7              JSR     SETUPT
        0D98    A0 06                LDY     #THGCHD
        0D9A    B1 E6                LDA     (ACC),Y
                            +        CMPBN   ARG1,L0DB7
                            +        DPUL    ACB
                            +        DPSH    ACB
        0DAC    A0 05                LDY     #THGSIB
        0DAE    B1 E4                LDA     (ACB),Y
        0DB0    A0 06                LDY     #THGCHD
        0DB2    91 E6                STA     (ACC),Y
        0DB4    4C 0DD2              JMP     L0DD2
        0DB7    20 16A7       L0DB7: JSR     SETUPT
        0DBA    A0 05                LDY     #THGSIB
        0DBC    B1 E6                LDA     (ACC),Y
                            +        CMPBN   ARG1,L0DB7
                            +        DPUL    ACB
                            +        DPSH    ACB
                            +        MOV     <(ACB),Y>,<<(ACC),Y>>
        0DD2                +L0DD2:  DPUL    ACC
        0DD8    A0 04                LDY     #THGPAR
                            +        MOV     <#$00>,<<(ACC),Y>>
        0DDE    C8                   INY                                  ; to THGSIB
        0DDF    91 E6                STA     (ACC),Y
        0DE1    60                   RTS

                                     PAGE
```

```
0DE2     A5 82         OPPRTN: LDA      ARG1                    ; print thing name
0DE4     20 16A7       LODE4:  JSR      SETUPT                  ; set up pointer to thing

0DE7     A0 07                 LDY      #THGPRP                 ; get address of thing's property list
                     +         MOV      <(ACC),Y>,ACB+1
0DED     C8                    INY
                     +         MOV      <(ACC),Y>,ACB
                     +         DMOV     ACB,ACC

                     +         DINC     ACC                     ; skip name length byte

0E00     20 17B8               JSR      SETAXB                  ; set AUX to point to it
0E03     4C 18B4               JMP      PRNTST                  ; and print it and return

                               PAGE
```

```


 0E06                 +OPRTN:  DMOV    STKPSV,STKPNT           ; restore pre-call stack pointer, count
                      +        MOV     STKCSV,STKCNT

 0E12     20 1720              JSR     PULLWD                  ; are there any local variables to restore?
 0E15     A5 E6                LDA     ACC
 0E17     F0 33                BEQ     L0E4C                   ; no, skip it

                      +        DMOVI   LOCVAR-2,ACB            ; yes, calc. addr. of last var to restore
                      +        MOV     ACC,ACD
 0E25     0A                   ASL     A
                      +        DADDB2  ACB

 0E2F     20 1720      L0E2F:  JSR     PULLWD                  ; pull the value of the var
 0E32     A0 01                LDY     #$01                    ; store it in the var
                      +        MOV     ACC,<<(ACB),Y>>
 0E38     88                   DEY
                      +        MOV     ACC+1,<<(ACB),Y>>
                      +        DDEC2   ACB                     ; decrement the var pointer
                      +        DECBN   ACD,L0E2F               ; and the count and loop if more to do

 0E4C     20 1720      L0E4C:  JSR     PULLWD                  ; pull the PC log. page
                      +        DMOV    ACC,PRGLPG

 0E57     20 1720              JSR     PULLWD                  ; pull the stack pointer save
                      +        DMOV    ACC,STKPSV

 0E62     20 1720              JSR     PULLWD                  ; pull the stack count save and PC
                      +        MOV     ACC+1,PRGIDX            ; low byte
                      +        MOV     ACC,STKCSV

                      +        MOV     <#$00>,PRGUPD           ; indicate need to locate new page

                      +        DMOV    ARG1,ACC                ; store the return value and return.
 0E79     4C 0B32              JMP     PTVRP1

                               PAGE
```

```

                                  ; jump to address ARG1

     0E7C                      +OPJUMP: DMOV     ARG1,ACC               ; setup to jump into middle of
                               +        DDEC     ACC                    ; predicate routine
     0E8F    4C 0BDA                    JMP      L0BDA                  ; and do it!


     0E92                      +OPPSW:  DMOV     ARG1,ACC               ; set AUX to point to string at
     0E9A    20 17C9                    JSR      SETAXW                 ;   word address
     0E9D    4C 18B4            L0E9D:  JMP      PRNTST                 ; and print it!


     0EA0    A5 82              OPMOVE: LDA      ARG1                   ; get number of first variable
     0EA2    20 0AC2                    JSR      GTVRA1                 ; get its contents
     0EA5    4C 0B32                    JMP      PTVRP1                 ; store into another variable


     0EA8                      +OPNOT:  D1COMP   ARG1,ACC
     0EB4    4C 0B32                    JMP      PTVRP1

     0EB7                      +L0EB7:  DMOV     ARG1,ACC
                               +        DMOV     ARG2,ACB
     0EC7    20 16DE                    JSR      L16DE
     0ECA    90 44                      BCC      L0F10
     0ECC    4C 0B8D                    JMP      PREDFL

     0ECF                      +L0ECF:  DMOV     ARG1,ACB
                               +        DMOV     ARG2,ACC
     0EDF    20 16DE                    JSR      L16DE
     0EE2    90 2C                      BCC      L0F10
     0EE4    4C 0B8D                    JMP      PREDFL

     0EE7    20 0D60            OPDECB: JSR      OPDEC
                               +        DMOV     ARG2,ACB
     0EF2    4C 0F08                    JMP      L0F08

     0EF5    20 0D43            OPINCB: JSR      OPINC
                               +        DMOV     ACC,ACB
                               +        DMOV     ARG2,ACC
     0F08    20 16DE            L0F08:  JSR      L16DE
                               +        JCS      PREDFL
     0F10    4C 0B84            L0F10:  JMP      PREDTR

     0F13    A5 82              OPTINT: LDA      ARG1
     0F15    20 16A7                    JSR      SETUPT
     0F18    A0 04                      LDY      #$04
     0F1A    A5 84                      LDA      ARG2
                               +        CMPBE    <(ACC),Y>,L0F10
     0F20    4C 0B8D                    JMP      PREDFL

     0F23                      +L0F23:  MOV      ARG2+1,ACC+1
     0F27    25 83                      AND      ARG1+1
     0F29    85 E5                      STA      ACB+1
                               +        MOV      ARG2,ACC
```

```
0F2F    25 82                   AND     ARG1
0F31    85 E4                   STA     ACB
0F33    20 16E9                 JSR     L16E9
0F36    F0 D8                   BEQ     L0F10
0F38    4C 0B8D                 JMP     PREDFL

0F3B                  +OPOR:    DOR     ARG2,ARG1,ACC
0F47    4C 0B32                 JMP     PTVRP1

0F4A                  +OPAND:   DAND    ARG2,ARG1,ACC
0F56    4C 0B32                 JMP     PTVRP1

                                PAGE
                                ; test attribute bit ARG2 of thing ARG1

        0F59    20 1629         OPTSTA: JSR     SETUPA
        0F5C    A5 E5                   LDA     ACB+1
        0F5E    25 E9                   AND     ACD+1
        0F60    85 E5                   STA     ACB+1
        0F62    A5 E4                   LDA     ACB
        0F64    25 E8                   AND     ACD
        0F66    05 E5                   ORA     ACB+1
        0F68    D0 A6                   BNE     L0F10
        0F6A    4C 0B8D                 JMP     PREDFL


                                ; set attribute bit ARG2 of thing ARG1

        0F6D    20 1629         OPSETA: JSR     SETUPA
        0F70    A0 01                   LDY     #$01
        0F72    A5 E4                   LDA     ACB
        0F74    05 E8                   ORA     ACD
        0F76    91 E6                   STA     (ACC),Y
        0F78    88                      DEY
        0F79    A5 E5                   LDA     ACB+1
        0F7B    05 E9                   ORA     ACD+1
        0F7D    91 E6                   STA     (ACC),Y
        0F7F    60                      RTS


                                ; clear attribute bit ARG2 of thing ARG1

        0F80    20 1629         OPCLRA: JSR     SETUPA
        0F83    A0 01                   LDY     #$01
        0F85    A5 E8                   LDA     ACD
        0F87    49 FF                   EOR     #$FF
        0F89    25 E4                   AND     ACB
        0F8B    91 E6                   STA     (ACC),Y
        0F8D    88                      DEY
        0F8E    A5 E9                   LDA     ACD+1
        0F90    49 FF                   EOR     #$FF
        0F92    25 E5                   AND     ACB+1
        0F94    91 E6                   STA     (ACC),Y
        0F96    60                      RTS

                                        PAGE
```

```

0F97                  +LOF97:   DMOV     ARG2,ACC
0F9F      A5 82                 LDA      ARG1
0FA1      4C 0AC9      LOFA1:   JMP      PTVRA1

0FA4      20 0D81      OPMOVT:  JSR      OPDSTT
0FA7      A5 82                 LDA      ARG1
0FA9      20 16A7               JSR      SETUPT
                      +         DPSH     ACC
0FB2      A0 04                 LDY      #THGPAR
                      +         MOV      ARG2,<<(ACC),Y>>
0FB8      20 16A7               JSR      SETUPT
0FBB      A0 06                 LDY      #THGCHD
0FBD      B1 E6                 LDA      (ACC),Y
0FBF      AA                    TAX
                      +         MOV      ARG1,<<(ACC),Y>>
                      +         DPUL     ACC
0FCA      8A                    TXA
0FCB      F0 04                 BEQ      LOFD1
0FCD      A0 05                 LDY      #THGSIB
0FCF      91 E6                 STA      (ACC),Y
0FD1      60           LOFD1:   RTS

0FD2                  +OPGTWD:  DASL     ARG2
                      +         DADD     ARG2,ARG1,ACC
0FE3      20 17B8               JSR      SETAXB
0FE6      20 17DB               JSR      FTAXWD
0FE9      4C 0B32               JMP      PTVRP1

0FEC                  +OPGTBY:  DADD     ARG2,ARG1,ACC
0FF9      20 17B8               JSR      SETAXB
0FFC      20 17E8               JSR      FTAXBA
0FFF      85 E6                 STA      ACC
                      +         MOV      <#$00>,ACC+1
1005      4C 0B32               JMP      PTVRP1

                                PAGE
```

```

                          ; get property ARG2 of thing ARG1

 1008    20 1669         OPGTP:  JSR     SETUPP
 100B    20 168E         L100B:  JSR     GTPNUM
                        +        CMPBE   ARG2,L103B
                        +        JSRCS   ADVPPT,L100B
 101A    A0 0B                   LDY     #HDRTHG+1
 101C    18                      CLC
 101D    B1 BA                   LDA     (FRZMEM),Y
 101F    65 BA                   ADC     FRZMEM
 1021    85 E4                   STA     ACB
 1023    88                      DEY
 1024    B1 BA                   LDA     (FRZMEM),Y
 1026    65 BB                   ADC     FRZMEM+1
 1028    85 E5                   STA     ACB+1
 102A    A5 84                   LDA     ARG2
 102C    0A                      ASL     A
 102D    A8                      TAY
 102E    88                      DEY
                        +        MOV     <(ACB),Y>,ACC
 1033    88                      DEY
                        +        MOV     <(ACB),Y>,ACC+1
 1038    4C 0B32                 JMP     PTVRP1
 103B    20 1693         L103B:  JSR     GTPLEN
 103E    C8                      INY
                        +        CMPBE   <#$00>,L105E
                        +        CMPJSN  <#$01>,FATAL
                        +        MOV     <(ACC),Y>,ACB+1
 104E    C8                      INY
                        +        MOV     <(ACC),Y>,ACB
                        +        DMOV    ACB,ACC
 105B    4C 0B32                 JMP     PTVRP1
 105E                   +L105E:  MOV     <(ACC),Y>,ACC
                        +        MOV     <#$00>,ACC+1
 1066    4C 0B32                 JMP     PTVRP1

                                 PAGE
```

```

                             ; get address of property ARG2 of thing ARG1

   1069     20 1669          OPGTPA: JSR      SETUPP
   106C     20 168E          L106C:  JSR      GTPNUM
                            +        CMPBE    ARG2,L107E
                            +        JCC      PTVRPZ
   1078     20 169D                  JSR      ADVPPT
   107B     4C 106C                  JMP      L106C
   107E                     +L107E:  DINC     ACC
   1084     18                       CLC
   1085     98                       TYA
   1086     65 E6                    ADC      ACC
   1088     85 E6                    STA      ACC
   108A     90 02                    BCC      L108E
   108C     E6 E7                    INC      ACC+1
   108E                     +L108E:  DSUB     ACC,FRZMEM,ACC
   109B     4C 0B32                  JMP      PTVRP1

                                     PAGE
```

```

                                ; get number of next property of thing ARG1 after property ARG2

        109E     20 1669        OPGTNP: JSR       SETUPP
        10A1     A5 84                  LDA       ARG2
        10A3     F0 12                  BEQ       L10B7
        10A5     20 168E        L10A5:  JSR       GTPNUM
                               +        CMPBE     ARG2,L10BD
                               +        JCC       PTVRPZ
        10B1     20 169D                JSR       ADVPPT
        10B4     4C 10A5                JMP       L10A5
        10B7     20 168E        L10B7:  JSR       GTPNUM
        10BA     4C 0B2C                JMP       PTVRPA
        10BD     20 169D        L10BD:  JSR       ADVPPT
        10C0     4C 10B7                JMP       L10B7

                                        PAGE
```

```

                          ; add ARG1 and ARG2

        10C3             +OPADD:  DADD    ARG1,ARG2,ACC
        10D0   4C 0B32            JMP     PTVRP1


                          ; subtract ARG2 from ARG1

        10D3             +OPSUB:  DSUB    ARG1,ARG2,ACC
        10E0   4C 0B32            JMP     PTVRP1


                          ; multiply ARG1 by ARG2

        10E3             +OPMUL:  DMOV    ARG1,ACC
                         +        DMOV    ARG2,ACB
        10F3   20 15FB            JSR     L15FB
        10F6   A5 E5              LDA     ACB+1
        10F8   D0 0A              BNE     L1104
        10FA   A5 E4              LDA     ACB
                         +        CMPBE   <#$02>,L1111
                         +        CMPBE   <#$04>,L110D
        1104   20 1568    L1104:  JSR     L1568
        1107   20 160A    L1107:  JSR     L160A
        110A   4C 0B32            JMP     PTVRP1
        110D             +L110D:  DASL    ACC
        1111             +L1111:  DASL    ACC
        1115   4C 1107            JMP     L1107

                                  PAGE
```

```

                              ; divide ARG1 by ARG2

     1118                    +OPDIV:  DMOV     ARG1,ACC
                             +        DMOV     ARG2,ACB
     1128     20 15FB                 JSR      L15FB
     112B     A5 E5                   LDA      ACB+1
     112D     D0 0A                   BNE      L1139
     112F     A5 E4                   LDA      ACB
                             +        CMPBE    <#$02>,L1143
                             +        CMPBE    <#$04>,L113F
     1139     20 15AD         L1139:  JSR      DIVIDE
     113C     4C 1107                 JMP      L1107
     113F                    +L113F:  DLSR     ACC
     1143                    +L1143:  DLSR     ACC
     1147     4C 1107                 JMP      L1107


                              ; get remainder of ARG1 divided by ARG2

     114A                    +OPRMD:  DMOV     ARG1,ACC
                             +        DMOV     ARG2,ACB
     115A     20 15FB                 JSR      L15FB
     115D     20 15AD                 JSR      DIVIDE
                             +        DMOV     ACB,ACC
     1168     4C 0B32                 JMP      PTVRP1

                                      PAGE
```

```

                                 ; test whether ARG1 is equal to any of the other args

  116B     A6 81         OPMTCH: LDX      ARGCNT
                        +        DXBNE    L1173
  1170     20 21D1               JSR      FATAL
  1173     A5 82          L1173: LDA      ARG1
                        +        CMPBN    ARG2,L117F
  1179     A5 83                 LDA      ARG1+1
                        +        CMPBE    ARG2+1,L11A0
  117F                  +L117F:  DXBEQ    L119D
  1182     A5 82                 LDA      ARG1
                        +        CMPBN    ARG3,L118E
  1188     A5 83                 LDA      ARG1+1
                        +        CMPBE    ARG3+1,L11A0
  118E                  +L118E:  DXBEQ    L119D
  1191     A5 82                 LDA      ARG1
                        +        CMPBN    ARG4,L1173
  1197     A5 83                 LDA      ARG1+1
                        +        CMPBE    ARG4+1,L11A0
  119D     4C 0B8D        L119D: JMP      PREDFL
  11A0     4C 0B84        L11A0: JMP      PREDTR

                                 PAGE
```

```

                                ; call procedure at addr. ARG1 and optionally pass ARG2, ARG3, and ARG4
                                ; as arguments

        11A3                   +OPCALL: DTS2BN  ARG1,L11B4              ; if argument 1 (call address/2) is
                               +        DMOVI   $0000,ACC               ; zero, just put zero in var
        11B1    4C 0B32                 JMP     PTVRP1                  ; these three lines could be replaced
                                                                        ; with "DTS2BE PTVRPZ"

        11B4                   +L11B4:  MOV     STKCSV,ACC              ; push the stack count save and low byte
                               +        MOV     PRGIDX,ACC+1            ; of the PC
        11BC    20 16F4                 JSR     PUSHWD

                               +        DMOV    STKPSV,ACC              ; push the stack pointer save
        11C7    20 16F4                 JSR     PUSHWD

                               +        DMOV    PRGLPG,ACC              ; push the PC logical page
        11D2    20 16F4                 JSR     PUSHWD

                               +        MOV     <#$00>,PRGUPD           ; indicate need to search for new page

                               +        DASL    ARG1,PRGIDX             ; make new PC := ARG1 * 2
        11E3    A9 00                   LDA     #$00
        11E5    2A                      ROL     A
        11E6    85 8C                   STA     PRGLPG+1

        11E8    20 173E                 JSR     FTPRBA                  ; get first byte of routine
        11EB    48                      PHA                             ; and save it

                               +        TSTABE  L1220                   ; if it's zero, no local variables

                                ; push the local variables the routine will use

        11F0    A2 00                   LDX     #$00
        11F2    48                L11F2: PHA
                               +        MOV     <LOCVAR,X>,ACC+1
        11F7    E8                      INX
                               +        MOV     <LOCVAR,X>,ACC
        11FC    CA                      DEX
        11FD    8A                      TXA
        11FE    48                      PHA
        11FF    20 16F4                 JSR     PUSHWD
        1202    20 173E                 JSR     FTPRBA
        1205    48                      PHA
        1206    20 173E                 JSR     FTPRBA
        1209    85 E6                   STA     ACC
                               +        PUL     ACC+1
        120E    68                      PLA
        120F    AA                      TAX
                               +        MOV     ACC+1,<<LOCVAR,X>>
        1214    E8                      INX
                               +        MOV     ACC,<<LOCVAR,X>>
        1219    E8                      INX
        121A    68                      PLA
                               +        SUB     ,<#$01>
        121E    D0 D2                   BNE     L11F2

        1220                  +L1220:   MOV     ARGCNT,ACD              ; do we pass any parameters?
                              +         DECBE   ACD,L124C               ; no

                              +         MOV     <#$00>,ACB              ; yes, copy them in
                              +         MOV     <#$00>,ACC
        1230    A6 E4          L1230:   LDX     ACB
        1232    B5 85                   LDA     ARG2+1,X
        1234    A6 E6                   LDX     ACC
        1236    95 9A                   STA     LOCVAR,X
        1238    E6 E6                   INC     ACC
        123A    A6 E4                   LDX     ACB
        123C    B5 84                   LDA     ARG2,X
        123E    A6 E6                   LDX     ACC
        1240    95 9A                   STA     LOCVAR,X
        1242    E6 E6                   INC     ACC
        1244    E6 E4                   INC     ACB
        1246    E6 E4                   INC     ACB

                              +         DECBN   ACD,L1230               ; loop if more paramaters to pass

        124C                  +L124C:   PUL     ACC                     ; get the first program byte again
        124F    20 16F4                 JSR     PUSHWD                  ; and push it so return can restore
                                                                        ; the local variables

                              +         MOV     STKCNT,STKCSV           ; save the stack pointer and count
                              +         DMOV    STKPNT,STKPSV

        125E    60                      RTS                             ; all done!

                                        PAGE
```

```

                                ; store word ARG3 at log. addr. ARG2 (offset) * 2 + ARG1 (base)
                                ; should have test to insure no overrun of end of frozen storage!

        125F    A5 84           OPPTWD: LDA     ARG2                    ; calculate logical address
        1261    0A                      ASL     A
        1262    26 85                   ROL     ARG2+1
        1264    18                      CLC
        1265    65 82                   ADC     ARG1
        1267    85 E6                   STA     ACC
        1269    A5 85                   LDA     ARG2+1
        126B    65 83                   ADC     ARG1+1
        126D    85 E7                   STA     ACC+1

                               +        DADD    ACC,FRZMEM,ACC          ; add base of frozen mem. to get phys. addr.

        127C    A0 00                   LDY     #$00                    ; store the word
                               +        MOV     ARG3+1,<<(ACC),Y>>
        1282    C8                      INY
                               +        MOV     ARG3,<<(ACC),Y>>

        1287    60                      RTS                             ; and return


                                ; store byte ARG3 at log. addr. ARG2 (offset) + ARG1 (base)
                                ; should have test to insure no overrun of end of frozen storage!

        1288    A5 84           OPPTBY: LDA     ARG2                    ; calculate logical address
        128A    18                      CLC
        128B    65 82                   ADC     ARG1
        128D    85 E6                   STA     ACC
        128F    A5 85                   LDA     ARG2+1
        1291    65 83                   ADC     ARG1+1
        1293    85 E7                   STA     ACC+1

                               +        DADD    ACC,FRZMEM,ACC          ; add base of frozen mem. to get phys. addr.

        12A2    A0 00                   LDY     #$00                    ; store the byte
                               +        MOV     ARG3,<<(ACC),Y>>

        12A8    60                      RTS                             ; and return

                                        PAGE
```

```

                             ; store ARG3 as property ARG2 of thing ARG1

12A9     20 1669       OPPTP:  JSR     SETUPP                   ; setup for thing property operations

12AC     20 168E       L12AC:  JSR     GTPNUM                   ; get the property number
                      +        CMPBE   ARG2,L12BE               ; if it is the one, go do it!

                      +        JSRCC   FATAL                    ; oops! past it!

12B8     20 169D               JSR     ADVPPT                   ; advance pointer
12BB     4C 12AC               JMP     L12AC                    ; and try again

                             ; got the property we wand

12BE     20 1693       L12BE:  JSR     GTPLEN                   ; get property length
12C1     C8                    INY
                      +        CMPBE   #$00,L12D7               ; if it is byte sized, go store it
                      +        CMPJSN  #$01,FATAL               ; if it isn't word sized, fatal error

                      +        MOV     ARG3+1,<<(ACC),Y>>       ; yes, store high byte
12D1     C8                    INY

                      +        MOV     ARG3,<<(ACC),Y>>         ; these two lines are unnecessary
12D6     60                    RTS

12D7                  +L12D7:  MOV     ARG3,<<(ACC),Y>>         ; store low byte
12DB     60                    RTS                              ; and return

                               PAGE
```

```

 12DC    20 1C8A      OPGTLN: JSR     OPPRST
                      +       DADD    ARG1,FRZMEM,ARG1
                      +       DADD    ARG2,FRZMEM,ARG2
 12F9    20 1D65              JSR     GETLIN
 12FC    85 E9                STA     ACD+1
                      +       MOV     <#$00>,ACD
 1302    A0 01                LDY     #$01
                      +       MOV     <#$00>,<<(ARG2),Y>>
                      +       MOV     <#$02>,LE0
                      +       MOV     <#$01>,LE1
 1310    A0 00         L1310: LDY     #00
 1312    B1 84                LDA     (ARG2),Y
 1314    C8                   INY
                      +       CMPRE   <(ARG2),Y>
                      +       DTSTRE  ACD
 1321    A5 E8                LDA     ACD
                      +       CMPJSE  <#$06>,L13BA
 132A    A5 E8                LDA     ACD
 132C    D0 2E                BNE     L135C
 132E    A0 06                LDY     #$06
 1330    A2 00                LDX     #$00
 1332                 +L1332: MOV     <#$00>,<<$D3,X>>
 1336    E8                   INX
                      +       DYBNE   L1332
 133A    A5 E1                LDA     LE1
 133C    A4 E0                LDY     LE0
 133E    C8                   INY
 133F    C8                   INY
 1340    C8                   INY
 1341    91 84                STA     (ARG2),Y
 1343    A4 E1                LDY     LE1
 1345    B1 82                LDA     (ARG1),Y
 1347    20 13F1              JSR     L13F1
 134A    B0 2E                BCS     L137A
 134C    A4 E1                LDY     LE1
 134E    B1 82                LDA     (ARG1),Y
 1350    20 13E0              JSR     L13E0
 1353    90 07                BCC     L135C
 1355    E6 E1                INC     LE1
 1357    C6 E9                DEC     ACD+1
 1359    4C 1310              JMP     L1310
 135C    A5 E9         L135C: LDA     ACD+1
 135E    F0 22                BEQ     L1382
 1360    A4 E1                LDY     LE1
 1362    B1 82                LDA     (ARG1),Y
 1364    20 13DA              JSR     L13DA
 1367    B0 19                BCS     L1382
 1369    A4 E1                LDY     LE1
 136B    B1 82                LDA     (ARG1),Y
 136D    A6 E8                LDX     ACD
 136F    95 D3                STA     INWORD,X
 1371    C6 E9                DEC     ACD+1
 1373    E6 E8                INC     ACD
 1375    E6 E1                INC     LE1
```

```
1377    4C 1310                 JMP     L1310
137A    85 D3         L137A:    STA     INWORD
137C    E6 E8                   INC     ACD
137E    C6 E9                   DEC     ACD+1
1380    E6 E1                   INC     LE1
1382    A5 E8         L1382:    LDA     ACD
1384    F0 8A                   BEQ     L1310
                     +          PSH     ACD+1
1389    A4 E0                   LDY     LE0
138B    C8                      INY
138C    C8                      INY
                     +          MOV     ACD,<<(ARG2),Y>>
1391    20 1A05                 JSR     CRNWRD
1394    20 141F                 JSR     L141F
1397    A4 E0                   LDY     LE0
                     +          MOV     ACB+1,<<(ARG2),Y>>
139D    C8                      INY
                     +          MOV     ACB,<<(ARG2),Y>>
13A2    C8                      INY
13A3    C8                      INY
13A4    C8                      INY
13A5    84 E0                   STY     LE0
13A7    A0 01                   LDY     #$01
                     +          ADD     <(ARG2),Y>,<#$01>,<<(ARG2),Y>>
                     +          PUL     ACD+1
                     +          MOV     <#$00>,ACD
13B7    4C 1310                 JMP     L1310

13BA    A5 E9         L13BA:    LDA     ACD+1
                     +          RTSEQ
13BF    A4 E1                   LDY     LE1
13C1    B1 82                   LDA     (ARG1),Y
13C3    20 13DA                 JSR     L13DA
                     +          RTSCS
13C9    E6 E1                   INC     LE1
13CB    C6 E9                   DEC     ACD+1
13CD    E6 E8                   INC     ACD
13CF    4C 13BA                 JMP     L13BA

13D2    20 2E 2C      SEPTAB: DB        ' .,?',CRCHAR,LFCHAR,TBCHAR,FFCHAR
13D5    3F 0D 0A
13D8    09 0C

13DA    20 13F1       L13DA:    JSR     L13F1
                     +          RTSCS
13E0    A0 00         L13E0:    LDY     #$00
13E2    A2 08                   LDX     #$08
13E4                 +L13E4:    CMPBE   <SEPTAB,Y>,L13EF
13E9    C8                      INY
                     +          DXBNE   L13E4
13ED    18            L13ED:    CLC
13EE    60                      RTS
13EF    38            L13EF:    SEC
13F0    60            L13F0:    RTS

13F1    48            L13F1:    PHA
```

```
13F2     20 1406                JSR     GTVCBA
13F5     A0 00                  LDY     #$00
13F7     B1 E6                  LDA     (ACC),Y
13F9     AA                     TAX
13FA     68                     PLA
13FB     F0 F0        L13FB:    BEQ     L13ED
13FD     C8                     INY
                     +          CMPBE   <(ACC),Y>,L13EF
1402     CA                     DEX
1403     4C 13FB                JMP     L13FB

1406     A0 08        GTVCBA:   LDY     #HDRVCB
                     +          MOV     <(FRZMEM),Y>,ACC+1
140C     C8                     INY
                     +          MOV     <(FRZMEM),Y>,ACC
                     +          DADD    ACC,FRZMEM,ACC
141E     60                     RTS

141F     20 1406      L141F:    JSR     GTVCBA
1422     A0 00                  LDY     #$00
1424     B1 E6                  LDA     (ACC),Y
1426     A8                     TAY
1427     C8                     INY
1428     B1 E6                  LDA     (ACC),Y
142A     0A                     ASL     A
142B     0A                     ASL     A
142C     0A                     ASL     A
142D     0A                     ASL     A
142E     85 E8                  STA     ACD
1430     C8                     INY
                     +          MOV     <(ACC),Y>,ACB+1
1435     C8                     INY
                     +          MOV     <(ACC),Y>,ACB
143A     C8                     INY
143B     98                     TYA
                     +          ADD     ,ACC,ACC
1441     90 02                  BCC     L1445
1443     E6 E7                  INC     ACC+1
1445     A0 00        L1445:    LDY     #$00
1447     4C 1450                JMP     L1450

144A     B1 E6        L144A:    LDA     (ACC),Y
                     +          CMPBG   PKWORD+1,L1470
1450                 +L1450:    DADDB1  ACC,ACD,ACC
                     +          DSUBB1  ACB,<#$10>,ACB
1466     A5 E5                  LDA     ACB+1
1468     30 06                  BMI     L1470
146A     D0 DE                  BNE     L144A
146C     A5 E4                  LDA     ACB
146E     D0 DA                  BNE     L144A
1470                 +L1470:    DSUBB1  ACC,ACD,ACC
                     +          DADDB1  ACB,<#$10>,ACB
1486     A5 E8                  LDA     ACD
1488     4A                     LSR     A
1489     4A                     LSR     A
148A     4A                     LSR     A
```

```
148B     4A                      LSR      A
148C     85 E8                   STA      ACD
148E     A0 00         L148E:    LDY      #$00
1490     A5 DB                   LDA      PKWORD+1
                      +          CMPBL    <(ACC),Y>,L14D0
1496     D0 1C                   BNE      L14B4
1498     C8                      INY
1499     A5 DA                   LDA      PKWORD
                      +          CMPBL    <(ACC),Y>,L14D0
149F     D0 13                   BNE      L14B4
14A1     A0 02                   LDY      #$02
14A3     A5 DD                   LDA      PKWORD+3
                      +          CMPBL    <(ACC),Y>,L14D0
14A9     D0 09                   BNE      L14B4
14AB     C8                      INY
14AC     A5 DC                   LDA      PKWORD+2
                      +          CMPBL    <(ACC),Y>,L14D0
14B2     F0 23                   BEQ      L14D7
14B4                  +L14B4:    DADDB1   ACC,ACD,ACC
                      +          DDEC     ACB
                      +          DTS2BN   ACB,L148E
14D0                  +L14D0:    MOV      <#$00>,<ACB+1,ACB>
14D6     60                      RTS
14D7                  +L14D7:    DSUB     ACC,FRZMEM,ACB
14E4     60                      RTS

                                 PAGE
```

```


                            ; print ASCII character ARG1

 14E5    A5 82               OPPRCH: LDA     ARG1
 14E7    4C 1B3F                     JMP     BFCHAR


                            ; print decimal number ARG1

 14EA                       +OPPRNM: DMOV    ARG1,ACC
 14F2    4C 14F5                     JMP     PRNTNM                  ; unnecessary


                            ; print decimal number in ACC

 14F5    A5 E7               PRNTNM: LDA     ACC+1                   ; negative?
                            +        JSRMI   L152E                   ;   yes, print '-' and negate
                            +        MOV     <#$00>,ACD              ; initialize digit count to 0
 1500                       +L1500:  DTSTBE  ACC,L1519               ; if the remainder is zero, print it now
                            +        DMOVI   $000A,ACB               ; set up divisor of 10
 150E    20 15AD                     JSR     DIVIDE                  ; divide
                            +        PSH     ACB                     ; push remainder onto stack
 1514    E6 E8                       INC     ACD                     ; increment digit count
 1516    4C 1500                     JMP     L1500                   ; do it again

 1519    A5 E8               L1519:  LDA     ACD                     ; is digit count zero?
 151B    F0 0C                       BEQ     L1529                   ;   yes, just print a '0' and return
 151D    68                  L151D:  PLA                             ; pull a digit off stack
                            +        ADD     ,<#'0'>                 ; convert to ASCII
 1521    20 1B3F                     JSR     BFCHAR                  ; print it
                            +        DECBN   ACD,L151D               ; decrement digit count, loop if more
 1528    60                          RTS                             ; return to caller

 1529    A9 30               L1529:  LDA     #'0'                    ; get code for '0'
 152B    4C 1B3F                     JMP     BFCHAR                  ; print it and return to caller

 152E    A9 2D               L152E:  LDA     #'-'                    ; get code for '-'
 1530    20 1B3F                     JSR     BFCHAR                  ; print it
 1533    4C 1611                     JMP     L1611                   ; negate the number, return

                                     PAGE
```

```
                         ; get a random number from 1 to ARG1

                                  IFF     RNGDBG

   1536                 +OPRNDM:  DMOV    ARG1,ACB             ; save range
   153E     20 21A0               JSR     L21A0                ; get the "random" number
   1541     20 15AD               JSR     DIVIDE               ; divide by range
                        +         DMOV    ACB,ACC              ; get the remainder
                        +         DINC    ACC                  ; increment it (base of result is 1)
   1552     4C 0B32               JMP     PTVRP1               ; and store it

                                  ENDIF

                         ; push ARG1 on stack

   1555                 +OPPUSH:  DMOV    ARG1,ACC
   155D     4C 16F4               JMP     PUSHWD


                         ; pull stack into variable ARG1

   1560     20 1720      OPPULL:  JSR     PULLWD
   1563     A5 82                 LDA     ARG1
   1565     4C 0FA1               JMP     L0FA1


   1568                 +L1568:   DPSH    ACD
                        +         DMOVI   $0000,ACD
   1576     A2 10                 LDX     #$10
   1578     A5 E4        L1578:   LDA     ACB
   157A     18                    CLC
   157B     29 01                 AND     #$01
   157D     F0 0C                 BEQ     L158B
                        +         DADC    ACC,ACD,ACD
   158B                 +L158B:   DROR    ACD
                        +         DROR    ACB
                        +         DXBNE   L1578
                        +         DMOV    ACB,ACC
                        +         DMOV    ACD,ACB
                        +         DPUL    ACD
   15AC     60                    RTS

                                  PAGE
```

```
                              ; divide ACC by ACB, quotient to ACC, remainder to ACB

        15AD                 +DIVIDE: DPSH     ACD
                             +        DMOV     ACC,ACD
                             +        DMOVI    $0000,ACC
        15C3    A2 11                 LDX      #$11
        15C5    38            L15C5:  SEC
        15C6    A5 E6                 LDA      ACC
        15C8    E5 E4                 SBC      ACB
        15CA    A8                    TAY
        15CB    A5 E7                 LDA      ACC+1
        15CD    E5 E5                 SBC      ACB+1
        15CF    90 05                 BCC      L15D6
        15D1    85 E7                 STA      ACC+1
        15D3    98                    TYA
        15D4    85 E6                 STA      ACC
        15D6                 +L15D6:  DROL     ACD
                             +        DROL     ACC
                             +        DXBNE    L15C5
        15E1    18                    CLC
                             +        DROR     ACC,ACB
                             +        DMOV     ACD,ACC
                             +        DPUL     ACD
        15FA    60                    RTS

        15FB                 +L15FB:  MOV      <#$00>,MDFLAG
        15FF    A5 E7                 LDA      ACC+1
        1601    20 161F               JSR      L161F
        1604    A5 E5                 LDA      ACB+1
        1606    20 161F               JSR      L161F
        1609    60                    RTS

        160A    A5 EA         L160A:  LDA      MDFLAG
        160C    29 01                 AND      #$01
                             +        RTSEQ
        1611    38            L1611:  SEC
        1612    A9 00                 LDA      #$00
        1614    E5 E6                 SBC      ACC
        1616    85 E6                 STA      ACC
        1618    A9 00                 LDA      #$00
        161A    E5 E7                 SBC      ACC+1
        161C    85 E7                 STA      ACC+1
        161E    60                    RTS

        161F                 +L161F:  TSTARP                        ; if positive, return
        1624    E6 EA                 INC      MDFLAG
        1626    4C 1611               JMP      L1611

                                      PAGE

                             ; setup stuff for thing attribute bit operations

        1629    A5 82        SETUPA: LDA     ARG1
        162B    20 16A7              JSR     SETUPT
        162E    A5 84                LDA     ARG2
                            +        CMPBL   <#$10>,L1643
                            +        SUB     ,<#$10>
                            +        DINC    ACC
                            +        DINC    ACC
        1643    85 E4        L1643:  STA     ACB
                            +        DMOVI   $0001,ACD
                            +        SUB     <#$0F>,ACB
        1652    AA                   TAX
        1653    F0 08        L1653:  BEQ     L165D
                            +        DASL    ACD
        1659    CA                   DEX
        165A    4C 1653              JMP     L1653
        165D    A0 00        L165D:  LDY     #$00
                            +        MOV     <(ACC),Y>,ACB+1
        1663    C8                   INY
                            +        MOV     <(ACC),Y>,ACB
        1668    60                   RTS


                             ; setup stuff for thing property operations

        1669    A5 82        SETUPP: LDA     ARG1
        166B    20 16A7              JSR     SETUPT
        166E    A0 07                LDY     #THGPRP
                            +        MOV     <(ACC),Y>,ACB+1
        1674    C8                   INY
                            +        MOV     <(ACC),Y>,ACB
                            +        DADD    ACB,FRZMEM,ACC
        1686    A0 00                LDY     #$00
        1688    B1 E6                LDA     (ACC),Y
        168A    0A                   ASL     A
        168B    A8                   TAY
        168C    C8                   INY
        168D    60                   RTS


                             ; get number of property pointed to by ACC

        168E    B1 E6        GTPNUM: LDA     (ACC),Y
        1690    29 1F                AND     #$1F
        1692    60                   RTS


                             ; get lenght of property pointed to by ACC

        1693    B1 E6        GTPLEN: LDA     (ACC),Y
                            +        REPT    5
                            +        ROR     A
                            +        ENDM
```

```
169A    29 07                AND     #$07
169C    60                   RTS


                     ; advance ACC to point to next property

169D    20 1693      ADVPPT: JSR     GTPLEN
16A0    AA                   TAX
16A1    C8           L16A1:  INY
                    +        DXBPL   L16A1
16A5    C8                   INY
16A6    60                   RTS


                     ; setup stuff for thing operations

16A7    85 E6        SETUPT: STA     ACC
                    +        MOV     <#$00>,ACC+1
16AD    A5 E6                LDA     ACC
                    +        REPT    3
                    +        DASL    ACC
                    +        ENDM
                    +        ADD     ,ACC
16BE    90 03                BCC     L16C3
16C0    E6 E7                INC     ACC+1
16C2    18                   CLC
16C3    69 35        L16C3:  ADC     #$35
16C5    85 E6                STA     ACC
16C7    90 02                BCC     L16CB
16C9    E6 E7                INC     ACC+1
16CB    A0 0B        L16CB:  LDY     #HDRTHG+1
                    +        ADD     <(FRZMEM),Y>,ACC,ACC
16D4    88                   DEY
16D5    B1 BA                LDA     (FRZMEM),Y
16D7    65 E7                ADC     ACC+1
16D9    65 BB                ADC     FRZMEM+1
16DB    85 E7                STA     ACC+1
16DD    60                   RTS

                             PAGE
```

```

16DE     A5 E5          L16DE:  LDA     ACB+1
16E0     45 E7                  EOR     ACC+1
16E2     10 05                  BPL     L16E9
16E4     A5 E5                  LDA     ACB+1
16E6     C5 E7                  CMP     ACC+1
16E8     60                     RTS
16E9     A5 E7          L16E9:  LDA     ACC+1
                       +        CMPBN   ACB+1,L16F3
16EF     A5 E6                  LDA     ACC
16F1     C5 E4                  CMP     ACB
16F3     60             L16F3:  RTS

16F4                   +PUSHWD: DDEC    STKPNT
16FF     A0 00                  LDY     #$00
                       +        MOV     ACC,<<(STKPNT),Y>>
                       +        DDEC    STKPNT
                       +        MOV     ACC+1,<<(STKPNT),Y>>
1714     E6 C8                  INC     STKCNT
1716     A5 C8                  LDA     STKCNT
                       +        CMPJSG  <#STCKMX>,FATAL
171F     60                     RTS

1720     A0 00          PULLWD: LDY     #$00
                       +        MOV     <(STKPNT),Y>,ACC+1
                       +        DINC    STKPNT
                       +        MOV     <(STKPNT),Y>,ACC
                       +        DINC    STKPNT
1736     C6 C8                  DEC     STKCNT
                       +        JSREQ   FATAL
173D     60                     RTS

                                PAGE
```

```

                               ; fetch next byte from PC into A

  173E     A5 8F          FTPRBA: LDA      PRGUPD                 ; need to find a new page?
  1740     F0 15                  BEQ      L1757                  ;   yes, go do it!

  1742     A4 8A                  LDY      PRGIDX                 ; get the byte
  1744     B1 8D                  LDA      (PRGMPT),Y

  1746     C8                     INY                             ; increment the low byte of the PC
  1747     84 8A                  STY      PRGIDX
                         +        RTSNE                           ; return unless we've entered a new page

  174C     A0 00                  LDY      #$00                   ; unnecessary!
  174E     84 8F                  STY      PRGUPD                 ; indicate new page
                         +        DINC     PRGLPG                 ; increment page number
  1756     60                     RTS                             ; return

  1757     A5 8C          L1757:  LDA      PRGLPG+1               ; is the page we're looking for frozen?
  1759     D0 06                  BNE      L1761
  175B     A5 8B                  LDA      PRGLPG
                         +        CMPBL    FRZPGS,L1778

  1761                   +L1761:  DMOV     PRGLPG,ACC             ; no, see if it is swapped in
  1769     20 1897                JSR      FNDPAG
  176C     85 90                  STA      PRGPPG                 ; save phys. page no.
  176E     B0 18                  BCS      L1788                  ; not found

                               ; we have the swappable page, fix up the pointers, etc.

  1770     20 1862        L1770:  JSR      MRKPAG                 ; indicate that we're using this page

  1773     18                     CLC                             ; add phys. page number to number
  1774     A5 90                  LDA      PRGPPG                 ; of frozen pages
  1776     65 BC                  ADC      FRZPGS

                               ; fix the memory pointers

  1778                   +L1778:  ADD      ,FRZMEM+1,PRGMPT+1     ; add base of frozen memory
                         +        MOV      <#$00>,PRGMPT

                         +        MOV      <#$FF>,PRGUPD          ; indicate that we have the page
  1785     4C 173E                JMP      FTPRBA                 ; and go get the byte

                               ; we need to load the page from disk

  1788                   +L1788:  CMPBN    AUXPPG,L1790           ; if we are about to load a new logical
                         +        MOV      <#$00>,AUXUPD          ; page into the physical page AUX points
                                                                  ; to, mark it as new page

  1790                   +L1790:  DMOV     SWPMEM,ACC             ; setup to read the page
                         +        ADD      PRGPPG,ACC+1,ACC+1
                         +        DMOV     PRGLPG,ACB

  17A7     20 1E0D                JSR      DRDBKF                 ; read the page (die if error)
```

```
                                                          
        17AA     A4 90                LDY      PRGPPG                    ; copy the new log. page number into
                                +     MOV      PRGLPG,<<(VMTAB1),Y>>     ; the VM table
                                +     MOV      PRGLPG+1,<<(VMTAB2),Y>>
                                                          
        17B4     98                   TYA
        17B5     4C 1770              JMP      L1770                     ; go fix up the pointers and fetch the byte
                                                          
                                      PAGE

                                ; set AUX to byte address in ACC

        17B8                    +SETAXB: MOV      ACC,AUXIDX
                                +        MOV      ACC+1,AUXLPG
                                +        MOV      <#$00>,AUXLPG+1
        17C4                    +L17C4:  MOV      <#$00>,AUXUPD
        17C8     60                      RTS


                                ; set AUX to word address in ACC

        17C9     A5 E6           SETAXW: LDA      ACC
        17CB     0A                      ASL      A
        17CC     85 93                   STA      AUXIDX
        17CE     A5 E7                   LDA      ACC+1
        17D0     2A                      ROL      A
        17D1     85 91                   STA      AUXLPG
        17D3     A9 00                   LDA      #$00
        17D5     2A                      ROL      A
        17D6     85 92                   STA      AUXLPG+1
        17D8     4C 17C4                 JMP      L17C4


                                ; fetch next word from AUX into ACC

        17DB     20 17E8         FTAXWD: JSR      FTAXBA
        17DE     48                      PHA
        17DF     20 17E8                 JSR      FTAXBA
        17E2     85 E6                   STA      ACC
                                +        PUL      ACC+1
        17E7     60                      RTS

                                         PAGE

                                ; fetch next byte from AUX into A

        17E8    A5 96           FTAXBA: LDA     AUXUPD                  ; need to find a new page?
        17EA    F0 15                   BEQ     L1801                   ;   yes, go do it!

        17EC    A4 93                   LDY     AUXIDX                  ; get the byte
        17EE    B1 94                   LDA     (AUXMPT),Y

        17F0    C8                      INY                             ; increment the low byte of AUX
        17F1    84 93                   STY     AUXIDX
                               +        RTSNE                           ; return unless we've entered a new page

        17F6    A0 00                   LDY     #$00                    ; unnecessary!
        17F8    84 96                   STY     AUXUPD                  ; indicate new page
                               +        DINC    AUXLPG                  ; increment page number
        1800    60                      RTS                             ; return

        1801    A5 92           L1801:  LDA     AUXLPG+1                ; is the page we're looking for frozen?
        1803    D0 06                   BNE     L180B
        1805    A5 91                   LDA     AUXLPG
        1807                   +L1807:  CMPBL   FRZPGS,L1822

        180B                   +L180B:  DMOV    AUXLPG,ACC              ; no, see if it is swapped in
        1813    20 1897                 JSR     FNDPAG
        1816    85 97                   STA     AUXPPG                  ; save phys. page no.
        1818    B0 18                   BCS     L1832                   ; not fount

                                ; we have the swappable page, fix up the pointers, etc.

        181A    20 1862         L181A:  JSR     MRKPAG                  ; indicate that we're using this page

        181D    18                      CLC                             ; add phys. page number to number of
        181E    A5 97                   LDA     AUXPPG                  ; frozen pages
        1820    65 BC                   ADC     FRZPGS

                                ; fix the memory pointers

        1822                   +L1822:  ADD     ,FRZMEM+1,AUXMPT+1      ; add base of memory
                               +        MOV     <#$00>,AUXMPT

                               +        MOV     <#$FF>,AUXUPD           ; indicate that we have the page
        182F    4C 17E8                 JMP     FTAXBA                  ; and go get the byte

                                ; we need to load the page from disk

        1832                   +L1832:  CMPBN   PRGPPG,L183A            ; if we are about to load a new logical
                               +        MOV     <#$00>,PRGUPD           ; page into the physical page the PC
                                                                        ; points to, mark it as a new page

        183A                   +L183A:  DMOV    SWPMEM,ACC              ; setup to read the page
                               +        ADD     AUXPPG,ACC+1,ACC+1
                               +        DMOV    AUXLPG,ACB

        1851    20 1E0D                 JSR     DRDBKF                  ; read the page (die if error)
```

```
     1854    A4 97               LDY     AUXPPG                  ; copy the new log. page number into
                         +       MOV     AUXLPG,<<(VMTAB1),Y>>   ; the VM table
                         +       MOV     AUXLPG+1,<<(VMTAB2),Y>>

     185E    98                  TYA
     185F    4C 181A             JMP     L181A                   ; go fix up the pointers and fetch the byte

                                 PAGE
```

```

                            ; we've just started using a new logical page, move it to front of our list
                            ; this make least recently used pages first candidates to be removed

 1862               +MRKPAG: CMPBE    MRUPAG,L1891
 1866     A6 BE              LDX      MRUPAG
 1868     85 BE              STA      MRUPAG
 186A     A8                 TAY
                    +        MOV      <(VMTAB3),Y>,ACC
 186F     8A                 TXA
 1870     91 C4              STA      (VMTAB3),Y
                    +        MOV      <(VMTAB4),Y>,ACC+1
                    +        MOV      <#$FF>,<<(VMTAB4),Y>>
 187A     A4 E7              LDY      ACC+1
                    +        MOV      ACC,<<(VMTAB3),Y>>
 1880     8A                 TXA
 1881     A8                 TAY
                    +        MOV      MRUPAG,<<(VMTAB4),Y>>
 1886     A5 E6              LDA      ACC
                    +        CMPBE    <#$FF>,L1892
 188C     A8                 TAY
                    +        MOV      ACC+1,<<(VMTAB4),Y>>
 1891     60         L1891:  RTS
 1892               +L1892:  MOV      ACC+1,LRUPAG
 1896     60                 RTS


                            ; search virtual memory table for logical page # in ACC

 1897     A6 BD      FNDPAG: LDX      SWPPGS
 1899     A0 00              LDY      #$00
 189B     A5 E6              LDA      ACC
 189D               +L189D:  CMPBN    <(VMTAB1),Y>,L18A9
 18A1     A5 E7              LDA      ACC+1
                    +        CMPBE    <(VMTAB2),Y>,L18B1
 18A7     A5 E6              LDA      ACC
 18A9     C8         L18A9:  INY
                    +        DXBNE    L189D
 18AD     A5 BF              LDA      LRUPAG
 18AF     38                 SEC
 18B0     60                 RTS
 18B1     98         L18B1:  TYA
 18B2     18                 CLC
 18B3     60                 RTS

                             PAGE
```

```

                          ; print string at AUX

     18B4                +PRNTST: MOV     <#$00>,<PRMMOD,PNYBCN>
                         +        MOV     <#$FF>,TMPMOD
     18BE    20 19B9      DONEXT: JSR     GETNYB
                         +        RTSCS
     18C4    85 E8                STA     ACD
     18C6    F0 48                BEQ     DOSPAC
                         +        CMPBL   <#$04>,DOSBWD
                         +        CMPBL   <#$06>,NEWMOD
     18D0    20 19AD              JSR     TSTMOD
                         +        TSTABN  L18E2
     18D7    A9 5B                LDA     #$5B
     18D9                +L18D9:  ADD     ,ACD
     18DC    20 1B3F      L18DC:  JSR     BFCHAR
     18DF    4C 18BE              JMP     DONEXT
     18E2                +L18E2:  CMPBN   <#$01>,DOSPCL
     18E6    A9 3B                LDA     #$3B
     18E8    4C 18D9              JMP     L18D9

     18EB                +DOSPCL: SUB     ACD,<#$07>
     18F0    90 0A                BCC     DOASCI
     18F2    F0 21                BEQ     DOCRLF
     18F4    A8                   TAY
     18F5    88                   DEY
     18F6    B9 1995              LDA     SPCLCH,Y
     18F9    4C 18DC              JMP     L18DC

     18FC    20 19B9      DOASCI: JSR     GETNYB
                         +        REPT    5
                         +        ASL     A
                         +        ENDM
     1904    48                   PHA
     1905    20 19B9              JSR     GETNYB
     1908    85 E8                STA     ACD
     190A    68                   PLA
     190B    05 E8                ORA     ACD
     190D    4C 18DC              JMP     L18DC

                                  PAGE
```

```

  1910    A9 20      DOSPAC: LDA     #' '
  1912    4C 18DC            JMP     L18DC

  1915    A9 0D      DOCRLF: LDA     #CRCHAR
  1917    20 1B3F            JSR     BFCHAR
  191A    A9 0A              LDA     #LFCHAR
  191C    4C 18DC            JMP     L18DC

  191F              +NEWMOD: SUB     ,<#$03>
  1922    A8                 TAY
  1923    20 19AD            JSR     TSTMOD
  1926    D0 05              BNE     L192D
  1928    84 CE              STY     TMPMOD
  192A    4C 18BE            JMP     DONEXT
  192D    84 CF      L192D:  STY     PRMMOD
                    +        CMPBE   PRMMOD,L1937
  1933    A0 00              LDY     #$00
  1935    84 CF              STY     PRMMOD
  1937    4C 18BE    L1937:  JMP     DONEXT

                             PAGE
```

```

 193A    00              L193A:  DB      $00

 193B                   +DOSBWD: DECA
                        +        REPT    6
                        +        ASL     A
                        +        ENDM
 1944    8D 193A                 STA     L193A
 1947    20 19B9                 JSR     GETNYB
 194A    0A                      ASL     A
 194B    69 01                   ADC     #$01
 194D    6D 193A                 ADC     L193A
 1950    A8                      TAY
                        +        MOV     <(SBWDPT),Y>,ACC
 1955    88                      DEY
                        +        MOV     <(SBWDPT),Y>,ACC+1
                        +        PSH     <PRMMOD,PNYBCN>
                        +        DPSH    PNYBBF
                        +        PSH     AUXIDX
                        +        DPSH    AUXLPG
 196F    20 17C9                 JSR     SETAXW
 1972    20 18B4                 JSR     PRNTST
                        +        DPUL    AUXLPG
                        +        PUL     AUXIDX
                        +        MOV     <#$00>,AUXUPD
                        +        DPUL    PNYBBF
                        +        PUL     <PNYBCN,PRMMOD>
                        +        MOV     <#$FF>,TMPMOD
 1992    4C 18BE                 JMP     DONEXT

 1995    30 31 32        SPCLCH: DB      '0123456789.,!?_#''"/\-:()'
 1998    33 34 35
 199B    36 37 38
 199E    39 2E 2C
 19A1    21 3F 5F
 19A4    23 27 22
 19A7    2F 5C 2D
 19AA    3A 28 29

 19AD    A5 CE           TSTMOD: LDA     TMPMOD
 19AF    10 03                   BPL     L19B4
 19B1    A5 CF                   LDA     PRMMOD
 19B3    60                      RTS
 19B4    A0 FF           L19B4:  LDY     #$FF
 19B6    84 CE                   STY     TMPMOD
 19B8    60                      RTS

                                 PAGE

 19B9    A5 D0        GETNYB: LDA     PNYBCN
 19BB    10 02                BPL     L19BF
 19BD    38                   SEC
 19BE    60                   RTS
 19BF    D0 15        L19BF:  BNE     L19D6
 19C1    E6 D0                INC     PNYBCN
 19C3    20 17DB              JSR     FTAXWD
                     +        DMOV    ACC,PNYBBF
 19CE    A5 D2                LDA     PNYBBF+1
 19D0    4A                   LSR     A
 19D1    4A                   LSR     A
 19D2    29 1F                AND     #$1F
 19D4    18                   CLC
 19D5    60                   RTS
 19D6                +L19D6:  DECABN  L19F3
                     +        MOV     <#$02>,PNYBCN
 19DF    A5 D2                LDA     PNYBBF+1
 19E1    4A                   LSR     A
 19E2    A5 D1                LDA     PNYBBF
 19E4    6A                   ROR     A
 19E5    A8                   TAY
 19E6    A5 D2                LDA     PNYBBF+1
 19E8    4A                   LSR     A
 19E9    4A                   LSR     A
 19EA    98                   TYA
 19EB    6A                   ROR     A
 19EC    4A                   LSR     A
 19ED    4A                   LSR     A
 19EE    4A                   LSR     A
 19EF    29 1F                AND     #$1F
 19F1    18                   CLC
 19F2    60                   RTS
 19F3                +L19F3:  MOV     <#$00>,PNYBCN
 19F7    A5 D2                LDA     PNYBBF+1
 19F9    10 04                BPL     L19FF
                     +        MOV     <#$FF>,PNYBCN
 19FF    A5 D1        L19FF:  LDA     PNYBBF
 1A01    29 1F                AND     #$1F
 1A03    18                   CLC
 1A04    60                   RTS

                              PAGE
```

```

                                ; crunch word to compare with vocab table entries

        1A05    A2 00           CRNWRD: LDX     #$00
        1A07    A0 06                   LDY     #$06
        1A09                    +L1A09: MOV     <#$05>,<<PKWORD,X>>
        1A0D    E8                      INX
                                +       DYBNE   L1A09
                                +       MOV     <#$06>,ACD+1
                                +       MOV     <#$00>,<ACB,ACC>
        1A1B    A6 E6           L1A1B:  LDX     ACC
        1A1D    E6 E6                   INC     ACC
                                +       MOV     <INWORD,X>,ACD
        1A23    D0 05                   BNE     L1A2A
        1A25    A9 05                   LDA     #$05
        1A27    4C 1A52                 JMP     L1A52
        1A2A    A5 E8           L1A2A:  LDA     ACD
        1A2C    20 1AAB                 JSR     TSTCHR
                                +       TSTABE  L1A43
                                +       ADD     ,<#$03>
        1A36    A6 E4                   LDX     ACB
        1A38    95 DA                   STA     PKWORD,X
        1A3A    E6 E4                   INC     ACB
                                +       DECJE   ACD+1,L1ACA
        1A43    A5 E8           L1A43:  LDA     ACD
        1A45    20 1AAB                 JSR     TSTCHR
                                +       DECABP  L1A62
                                +       SUB     ACD,<#$5B>
        1A52    A6 E4           L1A52:  LDX     ACB
        1A54    95 DA                   STA     PKWORD,X
        1A56    E6 E4                   INC     ACB
                                +       DECJN   ACD+1,L1A1B
        1A5F    4C 1ACA                 JMP     L1ACA
        1A62    D0 08           L1A62:  BNE     L1A6C
                                +       SUB     ACD,<#$3B>
        1A69    4C 1A52                 JMP     L1A52
        1A6C    A5 E8           L1A6C:  LDA     ACD
        1A6E    20 1A99                 JSR     L1A99
        1A71    D0 DF                   BNE     L1A52
        1A73    A9 06                   LDA     #$06
        1A75    A6 E4                   LDX     ACB
        1A77    95 DA                   STA     PKWORD,X
        1A79    E6 E4                   INC     ACB
                                +       DECBE   ACD+1,L1ACA
        1A7F    A5 E8                   LDA     ACD
                                +       REPT    5
                                +       LSR     A
                                +       ENDM
        1A86    29 03                   AND     #$03
        1A88    A6 E4                   LDX     ACB
        1A8A    95 DA                   STA     PKWORD,X
        1A8C    E6 E4                   INC     ACB
                                +       DECBE   ACD+1,L1ACA
        1A92    A5 E8                   LDA     ACD
        1A94    29 1F                   AND     #$1F
```

```
1A96    4C 1A52                 JMP     L1A52

1A99    A2 24         L1A99:    LDX     #$24
1A9B                 +L1A9B:    CMPBE   <SPCLCH,X>,L1AA6
                     +          DXBPL   L1A9B
1AA3    A0 00                   LDY     #$00
1AA5    60                      RTS
1AA6    8A            L1AA6:    TXA
                     +          ADD     ,<#$08>
1AAA    60                      RTS

1AAB                 +TSTCHR:   CMPBL   <#'a'>,L1AB6
                     +          CMPBG   <#'z'+1>,L1AB6
1AB3    A9 00                   LDA     #$00
1AB5    60                      RTS
1AB6                 +L1AB6:    CMPBL   <#'A'>,L1AC1
                     +          CMPBG   <#'Z'+1>,L1AC1
1ABE    A9 01                   LDA     #$01
1AC0    60                      RTS
1AC1                 +L1AC1:    TSTABE  L1AC9
1AC5    30 02                   BMI     L1AC9
1AC7    A9 02                   LDA     #$02
1AC9    60            L1AC9:    RTS

1ACA    A5 DB         L1ACA:    LDA     PKWORD+1
                     +          REPT    4
                     +          ASL     A
                     +          ENDM
1AD0    26 DA                   ROL     PKWORD
1AD2    0A                      ASL     A
1AD3    26 DA                   ROL     PKWORD
1AD5    A6 DA                   LDX     PKWORD
1AD7    86 DB                   STX     PKWORD+1
1AD9    05 DC                   ORA     PKWORD+2
1ADB    85 DA                   STA     PKWORD
1ADD    A5 DE                   LDA     LDE
                     +          REPT    4
                     +          ASL     A
                     +          ENDM
1AE3    26 DD                   ROL     PKWORD+3
1AE5    0A                      ASL     A
1AE6    26 DD                   ROL     PKWORD+3
1AE8    A6 DD                   LDX     PKWORD+3
1AEA    86 DD                   STX     PKWORD+3
1AEC    05 DF                   ORA     LDF
1AEE    85 DC                   STA     PKWORD+2
1AF0    A5 DD                   LDA     PKWORD+3
1AF2    09 80                   ORA     #$80
1AF4    85 DD                   STA     PKWORD+3
1AF6    60                      RTS

                                PAGE
```

```

                            ; init output routine and screen window

 1AF7                      +INITSC: MOV     <#$C1>,PRCSWL+1

                                    IFF     RNGDBG                  ; if RNG debug, save 2 lines at top!
                           +        MOV     <#$01>,WNDTOP
                                    ENDIF

                           +        MOV     <#$00>,<WNDLFT,L1BA0>
                           +        MOV     <#$28>,WNDWDT
                           +        MOV     <#$18>,WNDBOT
                           +        MOV     <#$BE>,PROMPT
                           +        MOV     <#$FF>,INVFLG

                            ; clear the screen

 1B16     20 FC58           CLRSCR: JSR     HOME
                           +        MOV     WNDTOP,LINCNT
 1B1D     60                        RTS


                            ; find the highest usable page of memory

 1B1E                      +FNDMEM: DMOVI2  LSTFLC+$0100,ACC
 1B26     A0 00                     LDY     #$00
 1B28     C6 E7             L1B28:  DEC     ACC+1
 1B2A     B1 E6                     LDA     (ACC),Y
                           +        CMPBN   <(ACC),Y>,L1B28
 1B30     49 FF                     EOR     #$FF
 1B32     91 E6                     STA     (ACC),Y
                           +        CMPBN   <(ACC),Y>,L1B28
 1B38     49 FF                     EOR     #$FF
 1B3A     91 E6                     STA     (ACC),Y
 1B3C     A5 E7                     LDA     ACC+1
 1B3E     60                        RTS

                                    PAGE
```

```

                                 ; buffer a character for output

         1B3F     A6 EB          BFCHAR: LDX     CHRPTR                  ; get buffer pointer

                                +        CMPJE   <#CRCHAR>,PRNTBF        ; if char is a CR, flush buffer
                                +        CMPBL   <#' '>,L1B61            ; if it is a control character, discard it
                                +        CMPBL   <#$60>,L1B57            ; if it is in 64 char subset, buffer it as is

                                         IFT     LC40
         1B50     24 32                  BIT     INVFLG                  ; if inverse, convert LC to UC
         1B52     30 03                  BMI     L1B57
                                         ENDIF

                                +        SUB     ,<#$20>                 ;   yes, convert to upper case

         1B57     09 80          L1B57:  ORA     #$80                    ; set high bit for Apple

         1B59     9D 0200                STA     BUFFER,X                ; store it in buffer
                                +        CPXBG   WNDWDT,L1B64            ; if buffer is full, print some of it

         1B60     E8                     INX                             ; increment pointer
         1B61     86 EB          L1B61:  STX     CHRPTR                  ; save pointer
         1B63     60                     RTS                             ; return

                                 ; find last space in buffer, if any

         1B64     A9 A0          L1B64:  LDA     #" "                    ; load a space for comparison

         1B66                   +L1B66:  CMPBE   <BUFFER,X>,L1B70        ; if this is one, we've got it
                                +        DXBNE   L1B66                   ; no, loop if more characters in buffer

         1B6E     A6 21                  LDX     WNDWDT                  ; no space... use last character

         1B70     86 EC          L1B70:  STX     CHRPT2                  ; save pointer
         1B72     86 EB                  STX     CHRPTR

         1B74     20 1C10                JSR     PRNTBF                  ; print line up to this point

                                 ; move rest of line back to beginning of buffer

         1B77     E6 EC          L1B77:  INC     CHRPT2                  ; get pointer to next char
         1B79     A6 EC                  LDX     CHRPT2
                                +        CPXRGT  WNDWDT                  ; if it is past the last char, return

         1B82     BD 0200                LDA     BUFFER,X                ; get the character
         1B85     A6 EB                  LDX     CHRPTR                  ; get the pointer to the new loc
         1B87     9D 0200                STA     BUFFER,X                ; store the character there
         1B8A     E6 EB                  INC     CHRPTR                  ; and increment the pointer

         1B8C     A6 EC                  LDX     CHRPT2                  ; unnecessary!
         1B8E     4C 1B77                JMP     L1B77                   ; try for another one

                                         PAGE
```

```

                            ; output the buffer to the screen, and to the printer if enabled

     1B91    A0 11          OUTBUF: LDY     #HDRFLG+1
     1B93    B1 BA                  LDA     (FRZMEM),Y
     1B95    29 01                  AND     #$01
                           +        JSRNE   PRTBUF
     1B9C    20 1BF5                JSR     DSPBUF
     1B9F    60                     RTS


                            ; output the buffer to the printer

     1BA0    00             L1BA0:  DB      $00                     ; printer initialization flag

     1BA1                  +PRTBUF: DPSH    CSWL                    ; save our output vector
                           +        PSH     CURSRH                  ; and cursor column

                           +        DMOV    PRCSWL,CSWL             ; get vector for printer

     1BB2    A2 00                  LDX     #$00                    ; start with position 0 in buffer

     1BB4    AD 1BA0                LDA     L1BA0                   ; is printer initialized?
     1BB7    D0 1C                  BNE     L1BD5                   ; yes, go print it
     1BB9    EE 1BA0                INC     L1BA0                   ; no, but now it will be

     1BBC    A9 89                  LDA     #$89                    ; output ^I80N
     1BBE    20 FDED                JSR     COUT                    ;   (this sets printer width to 80
     1BC1    A9 91                  LDA     #$91                    ;    characters, thereby disabling
     1BC3    8D 0779                STA     PRTWDT                  ;    screen echo (we hope!))
     1BC6    A9 B8                  LDA     #$B8
     1BC8    20 FDED                JSR     COUT
     1BCB    A9 B0                  LDA     #$B0
     1BCD    20 FDED                JSR     COUT
     1BD0    A9 CE                  LDA     #$CE
     1BD2    20 FDED                JSR     COUT

     1BD5                  +L1BD5:  CPXBE   CHRPTR,L1BE3            ; are we done yet?

     1BD9    BD 0200                LDA     BUFFER,X                ; no, get character
     1BDC    20 FDED                JSR     COUT                    ; and output it

     1BDF    E8                     INX                             ; increment pointer
     1BE0    4C 1BD5                JMP     L1BD5                   ; and go for another one

     1BE3                  +L1BE3:  DMOV    CSWL,PRCSWL             ; save print vector again (may have changed)

                           +        PUL     CURSRH                  ; restore cursor column
                           +        DPUL    CSWL                    ; and display vector
     1BF4    60                     RTS                             ; and return


                            ; output the buffer to the display

     1BF5    A2 00          DSPBUF: LDX     #$00                    ; start with position 0 in buffer
```

```

1BF7                  +L1BF7:  CPXBE   CHRPTR,L1C05          ; are we done yet?

1BFB    BD 0200                LDA     BUFFER,X              ; get the character
1BFE    20 FDF0                JSR     COUT1                 ; and output it

1C01    E8                     INX                           ; increment pointer
1C02    4C 1BF7                JMP     L1BF7                 ; and go for another one

1C05    A2 00          L1C05:  LDX     #$00                  ; reset pointer to beginning
1C07    86 EB                  STX     CHRPTR
1C09    60                     RTS                           ; and return

                               PAGE
```

```
    1C0A    5B 4D 4F      MOREMS: DB       '[MORE]'
    1C0D    52 45 5D
    0006                  MRMSLN  EQU      *-MOREMS

    1C10    E6 ED         PRNTBF: INC      LINCNT
    1C12    A5 ED                 LDA      LINCNT
                         +        CMPBL    WNDBOT,L1C40
                         +        DMOVI    MOREMS,ACC
    1C20    A2 06                 LDX      #MRMSLN
                         +        MOV      <#$3F>,INVFLG
    1C26    20 1D57               JSR      SHWMSG
                         +        MOV      <#$FF>,INVFLG
    1C2D    20 FD0C               JSR      RDKEY
                         +        SUB      CURSRH,<#$06>,CURSRH
    1C37    20 FC9C               JSR      CLREOL
                         +        MOV      WNDTOP,LINCNT
    1C3E    E6 ED                 INC      LINCNT
    1C40                 +L1C40:  PSH      CHRPTR
    1C43    20 1B91               JSR      OUTBUF
    1C46    68                    PLA
                         +        CMPBE    WNDWDT,L1C50
    1C4B    A9 8D                 LDA      #$8D
    1C4D    20 FDF0               JSR      COUT1
    1C50    A0 11          L1C50: LDY      #HDRFLG+1
    1C52    B1 BA                 LDA      (FRZMEM),Y
    1C54    29 01                 AND      #$01
    1C56    F0 21                 BEQ      L1C79
                         +        DPSH     CSWL
                         +        DMOV     PRCSWL,CSWL
    1C66    A9 8D                 LDA      #$8D
    1C68    20 FDED               JSR      COUT
                         +        DMOV     CSWL,PRCSWL
                         +        DPUL     CSWL
    1C79    A2 00          L1C79: LDX      #$00
    1C7B    4C 1B61               JMP      L1B61

                                  PAGE
```

```

1C7E    53 43 4F      SCORMS: DB      'SCORE:'
1C81    52 45 3A
0006                  SCMSLN  EQU     *-SCORMS

1C84    54 49 4D      TIMEMS: DB      'TIME:'
1C87    45 3A
0005                  TMMSLN  EQU     *-TIMEMS

1C89    00            L1C89:  DB      $00

1C8A    20 1B91       OPPRST: JSR     OUTBUF                  ; print what's in the buffer
                     +        PSH     <CURSRH,CURSRV>         ; save the cursor position
                     +        MOV     <#$00>,<CURSRH,CURSRV>  ; home the cursor
1C99    20 FC22               JSR     VTAB
                     +        MOV     <#$3F>,INVFLG           ; set inverse mode

1CA0    A9 10                 LDA     #$10                    ; get gloval var 0
1CA2    20 0AC2               JSR     GTVRA1
1CA5    A5 E6                 LDA     ACC                     ; is it save as last time?
                     +        CMPBE   L1C89,L1CB8             ; yes, don't print it
1CAC    8D 1C89               STA     L1C89                   ; no, save for next time's compare
1CAF    20 0DE4               JSR     L0DE4                   ; output thing name
1CB2    20 1BF5               JSR     DSPBUF                  ; send it to display
1CB5    20 FC9C               JSR     CLREOL                  ; clear rest of line

1CB8                 +L1CB8:  MOV     <#$19>,CURSRH           ; tab over
1CBC    A5 F3                 LDA     STLTYP                  ; score or time?
1CBE    D0 1B                 BNE     L1CDB                   ; time
                     +        DMOVI   SCORMS,ACC              ; score, print "SCORE:"
1CC8    A2 06                 LDX     #SCMSLN
1CCA    20 1D57               JSR     SHWMSG
1CCD    E6 24                 INC     CURSRH                  ; one space
1CCF    A9 11                 LDA     #$11                    ; get global var 1 (score)
1CD1    20 0AC2               JSR     GTVRA1
1CD4    20 14F5               JSR     PRNTNM                  ; output it as decimal number
1CD7    A9 2F                 LDA     #'/'                    ; seperator
1CD9    D0 2A                 BNE     L1D05                   ; always taken

1CDB                 +L1CDB:  DMOVI   TIMEMS,ACC              ; print "TIME:"
1CE3    A2 05                 LDX     #TMMSLN
1CE5    20 1D57               JSR     SHWMSG
1CE8    E6 24                 INC     CURSRH                  ; one space
1CEA    A9 11                 LDA     #$11                    ; get global var 1 (time)
1CEC    20 0AC2               JSR     GTVRA1
1CEF    A5 E6                 LDA     ACC                     ; is it zero?
1CF1    D0 02                 BNE     L1CF5
1CF3    A9 18                 LDA     #$18                    ; yes, make it 24:00
1CF5                 +L1CF5:  CMPBM   <#$0C>,L1D00            ; is it A.M. or P.M.?
1CF9    F0 05                 BEQ     L1D00
1CFB    38                    SEC                             ; P.M., convert to 1-12 range
1CFC    E9 0C                 SBC     #$0C                    ;  by subtracting 12
1CFE    85 E6                 STA     ACC
1D00    20 14F5        L1D00: JSR     PRNTNM                  ; print out hours
1D03    A9 3A                 LDA     #':'
```

```
1D05    20 1B3F     L1D05:  JSR     BFCHAR          ; print the seperator
1D08    A9 12               LDA     #$12            ; get global var 2 (turns/minutes)
1D0A    20 0AC2             JSR     GTVRA1
1D0D    A5 F3               LDA     STLTYP          ; time?
1D0F    F0 2F               BEQ     L1D40           ; no, go print turns
1D11    A5 E6               LDA     ACC             ; yes, are minutes < 10?
                    +       CMPBG   <#$0A>,L1D1C    ; no
1D17    A9 B0               LDA     #$B0            ; yes, print a space (?)
1D19    20 1B3F             JSR     BFCHAR
1D1C    20 14F5     L1D1C:  JSR     PRNTNM          ; print the minutes
1D1F    A9 A0               LDA     #$A0            ; print a space
1D21    20 1B3F             JSR     BFCHAR
1D24    A9 11               LDA     #$11            ; get global var 1 (hours)
1D26    20 0AC2             JSR     GTVRA1
1D29    A5 E6               LDA     ACC             ; is it A.M. or P.M.?
                    +       CMPBP   <#$0C>,L1D33    ; P.M.
1D2F    A9 C1               LDA     #"A"            ; A.M.
1D31    D0 02               BNE     L1D35
1D33    A9 D0       L1D33:  LDA     #"P"
1D35    20 1B3F     L1D35:  JSR     BFCHAR          ; print the "A" or "P"
1D38    A9 CD               LDA     #"M"
1D3A    20 1B3F             JSR     BFCHAR          ; print the "M"
1D3D    4C 1D43             JMP     L1D43

1D40    20 14F5     L1D40:  JSR     PRNTNM          ; print the score
1D43    20 1BF5     L1D43:  JSR     DSPBUF          ; display the buffer
1D46    20 FC9C             JSR     CLREOL          ; clear out the line
                    +       MOV     <#$FF>,INVFLG   ; back to normal video mode
                    +       PUL     <CURSRV,CURSRH> ; and the old cursor loc
1D53    20 FC22             JSR     VTAB
1D56    60                  RTS                     ; return to caller

1D57    A0 00       SHWMSG: LDY     #$00
1D59    B1 E6       L1D59:  LDA     (ACC),Y
1D5B    09 80               ORA     #$80
1D5D    20 FDF0             JSR     COUT1
1D60    C8                  INY
                    +       DXBNE   L1D59
1D64    60                  RTS

                            PAGE
```

```

1D65     20 1B91      GETLIN: JSR      OUTBUF
                    +         MOV      WNDTOP,LINCNT
1D6C     20 FD6F              JSR      GETLN1
1D6F     E6 ED                INC      LINCNT
                    +         MOV      <#$8D>,<<BUFFER,X>>
1D76     E8                   INX
1D77     8A                   TXA
1D78     48                   PHA
1D79     A0 11                LDY      #HDRFLG+1
1D7B     B1 BA                LDA      (FRZMEM),Y
1D7D     29 01                AND      #$01
1D7F     F0 0A                BEQ      L1D8B
1D81     8A                   TXA
1D82     85 EB                STA      CHRPTR
1D84     20 1BA1              JSR      PRTBUF
                    +         MOV      <#$00>,CHRPTR
1D8B     68           L1D8B:  PLA
1D8C     A0 00                LDY      #$00
                    +         CMPBL    <(ARG1),Y>,L1D94
1D92     B1 82                LDA      (ARG1),Y
1D94     48           L1D94:  PHA
1D95     F0 1A                BEQ      L1DB1
1D97     AA                   TAX
1D98     B9 0200      L1D98:  LDA      BUFFER,Y
1D9B     29 7F                AND      #$7F
                    +         CMPBL    <#'A'>,L1DA7
                    +         CMPBG    <#'Z'+1>,L1DA7
1DA5     09 20                ORA      #$20
1DA7     C8           L1DA7:  INY
1DA8     91 82                STA      (ARG1),Y
                    +         CMPBE    <#CRCHAR>,L1DB1
                    +         DXBNE    L1D98
1DB1     68           L1DB1:  PLA
1DB2     60                   RTS

                              PAGE
```

```


1DB3    01             IOB:     DB     $01                          ; IOB type
1DB4    60             IOBSLT:  DB     $60                          ; Slot * 16
1DB5    01             IOBDRV:  DB     $01                          ; Drive
1DB6    00                      DB     $00                          ; Volume
1DB7    00             IOBTRK:  DB     $00                          ; Track
1DB8    00             IOBSCT:  DB     $00                          ; Sector
1DB9    1DC4                    DW     DCT                          ; Device Characteristics Table
1DBB    0000           IOBBUF:  DW     $0000                        ; I/O buffer
1DBD    0000                    DW     $0000                        ; unused
1DBF    00             IOBCMD:  DB     $00                          ; Command
1DC0    00                      DB     $00                          ; Status
1DC1    00                      DB     $00                          ; Actual volume
1DC2    60                      DB     $60                          ; Previous slot * 16
1DC3    01                      DB     $01                          ; Previous drive

1DC4    00 01 EF       DCT:     DB     $00,$01,$EF,$D8
1DC7    D8

1DC8    8D 1DBF        DISKIO:  STA    IOBCMD
                      +         DMOV   ACC,IOBBUF
                      +         MOV    #$03,IOBTRK
1DDA    A5 E4                   LDA    ACB
1DDC    A6 E5                   LDX    ACB+1
1DDE    38                      SEC
1DDF    E5 7F          L1DDF:   SBC    SECPTK
1DE1    B0 04                   BCS    L1DE7
                      +         DXBMI  L1DED
1DE6    38                      SEC
1DE7    EE 1DB7        L1DE7:   INC    IOBTRK
1DEA    4C 1DDF                 JMP    L1DDF
1DED                  +L1DED:   ADD    ,SECPTK,IOBSCT
1DF3    A9 1D                   LDA    #>IOB
1DF5    A0 B3                   LDY    #<IOB
1DF7    4C 2900                 JMP    RWTS

1DFA                  +DRDBUF:  DMOVI  BUFFER,ACC
1E02                  +DRDNXT:  DINC   ACB
1E08    A9 01          DRDBLK:  LDA    #$01
1E0A    4C 1DC8                 JMP    DISKIO

1E0D    20 1E08        DRDBKF:  JSR    DRDBLK
                      +         JSRCS  FATAL
1E15    60                      RTS

1E16                  +DWRBUF:  DMOVI  BUFFER,ACC
1E1E                  +DWRNXT:  DINC   ACB
1E24    A9 02                   LDA    #$02
1E26    4C 1DC8                 JMP    DISKIO

                                PAGE
```

```


        1E29     86 E8         OUTMSG: STX      ACD
        1E2B     A0 00                 LDY      #$00
        1E2D     84 E9                 STY      ACD+1
        1E2F     A4 E9         L1E2F:  LDY      ACD+1
        1E31     B1 E6                 LDA      (ACC),Y
        1E33     20 1B3F               JSR      BFCHAR
        1E36     E6 E9                 INC      ACD+1
                              +        DECBN    ACD,L1E2F
        1E3C     60                    RTS

        1E3D     50 4C 45      L1E3D:  DB       'PLEASE INSERT SAVE DISKETTE,'
        1E40     41 53 45
        1E43     20 49 4E
        1E46     53 45 52
        1E49     54 20 53
        1E4C     41 56 45
        1E4F     20 44 49
        1E52     53 4B 45
        1E55     54 54 45
        1E58     2C

        1E59     00            L1E59:  DB       $00

        1E5A     53 4C 4F      L1E5A:  DB       'SLOT     (1-7):'
        1E5D     54 20 20
        1E60     20 20 20
        1E63     28 31 2D
        1E66     37 29 3A
        1E69     36 31 38      L1E69:  DB       '618'

        1E6C     44 52 49      L1E6C:  DB       'DRIVE    (1-2):'
        1E6F     56 45 20
        1E72     20 20 20
        1E75     28 31 2D
        1E78     32 29 3A
        1E7B     32 31 33      L1E7B:  DB       '213'

        1E7E     50 4F 53      L1E7E:  DB       'POSITION (0-7):'
        1E81     49 54 49
        1E84     4F 4E 20
        1E87     28 30 2D
        1E8A     37 29 3A
        1E8D     30 30 38      L1E8D:  DB       '008'

        1E90     44 45 46      L1E90:  DB       'DEFAULT = '
        1E93     41 55 4C
        1E96     54 20 3D
        1E99     20

        1E9A     2D 2D 2D      L1E9A:  DB       '--- PRESS ''RETURN'' KEY TO BEGIN ---'
        1E9D     20 50 52
        1EA0     45 53 53
        1EA3     20 27 52
```

```
1EA6    45 54 55
1EA9    52 4E 27
1EAC    20 4B 45
1EAF    59 20 54
1EB2    4F 20 42
1EB5    45 47 49
1EB8    4E 20 2D
1EBB    2D 2D

                                PAGE
```

```
        1EBD    20 1B16     L1EBD:  JSR     CLRSCR
        1EC0    20 1C10             JSR     PRNTBF
        1EC3    20 1C10             JSR     PRNTBF
                           +        DMOVI   L1E3D,ACC
        1ECE    A2 1C               LDX     #$1C
        1ED0    20 1E29             JSR     OUTMSG
        1ED3    20 1C10             JSR     PRNTBF
                           +        MOV     <#$24>,L1E59
        1EDB    20 1F4C             JSR     L1F4C
        1EDE    8D 1E8D             STA     L1E8D
        1EE1    20 1B3F             JSR     BFCHAR
                           +        MOV     <#$00>,L1E59
        1EE9    20 1F4C             JSR     L1F4C
        1EEC    AA                  TAX
        1EED    29 07               AND     #$07
                           +        REPT    4
                           +        ASL     A
                           +        ENDM
        1EF3    8D 1DB4             STA     IOBSLT
        1EF6    8A                  TXA
        1EF7    8D 1E69             STA     L1E69
        1EFA    20 1B3F             JSR     BFCHAR
                           +        MOV     <#$12>,L1E59
        1F02    20 1F4C             JSR     L1F4C
        1F05    AA                  TAX
        1F06    29 03               AND     #$03
        1F08    8D 1DB5             STA     IOBDRV
        1F0B    8A                  TXA
        1F0C    8D 1E7B             STA     L1E7B
        1F0F    20 1B3F             JSR     BFCHAR
        1F12    20 1C10     L1F12:  JSR     PRNTBF
                           +        DMOVI   L1E9A,ACC
        1F1D    A2 23               LDX     #$23
        1F1F    20 1E29             JSR     OUTMSG
        1F22    20 1B91             JSR     OUTBUF
        1F25    20 FD0C             JSR     RDKEY
                           +        CMPBN   <#$8D>,L1F12
                           +        MOV     <#$FF>,<ACB,ACB+1>
        1F32    AD 1E8D             LDA     L1E8D
        1F35    29 07               AND     #$07
        1F37    F0 0F               BEQ     L1F48
        1F39    A8                  TAY
        1F3A               +L1F3A:  DADDB2  ACB,<#$40>
                           +        DYBNE   L1F3A
        1F48    20 1C10     L1F48:  JSR     PRNTBF
        1F4B    60                  RTS

                                    PAGE
```

```

1F4C     20 1C10     L1F4C:  JSR      PRNTBF
                     +       DMOVI    L1E5A,ACC
                     +       DADDB2   ACC,L1E59
1F63     A2 0F               LDX      #$0F
1F65     20 1E29             JSR      OUTMSG
1F68     20 1B91             JSR      OUTBUF
                     +       MOV      <#$19>,CURSRH
                     +       MOV      <#$3F>,INVFLG
                     +       DMOVI    L1E90,ACC
1F7B     A2 0A               LDX      #$0A
1F7D     20 1D57             JSR      SHWMSG
                     +       DMOVI    L1E69,ACC
                     +       DADDB2   ACC,L1E59
1F94     A2 01               LDX      #$01
1F96     20 1D57             JSR      SHWMSG
                     +       MOV      <#$FF>,INVFLG
1F9D     20 FD0C             JSR      RDKEY
1FA0     48                  PHA
                     +       MOV      <#$19>,CURSRH
1FA5     20 FC9C             JSR      CLREOL
1FA8     68                  PLA
1FA9     AC 1E59             LDY      L1E59
                     +       CMPBN    <#$8D>,L1FB3
1FB0     B9 1E69             LDA      L1E69,Y
1FB3     29 7F       L1FB3:  AND      #$7F
                     +       CMPBL    <L1E69+1,Y>,L1F4C
                     +       CMPBG    <L1E69+2,Y>,L1F4C
1FBF     60                  RTS

                             PAGE
```

```

 1FC0     50 4C 45       L1FC0:  DB       'PLEASE RE-INSERT GAME DISKETTE,'
 1FC3     41 53 45
 1FC6     20 52 45
 1FC9     2D 49 4E
 1FCC     53 45 52
 1FCF     54 20 47
 1FD2     41 4D 45
 1FD5     20 44 49
 1FD8     53 4B 45
 1FDB     54 54 45
 1FDE     2C

 1FDF     2D 2D 2D       L1FDF:  DB       '--- PRESS ''RETURN'' KEY TO CONTINUE ---'
 1FE2     20 50 52
 1FE5     45 53 53
 1FE8     20 27 52
 1FEB     45 54 55
 1FEE     52 4E 27
 1FF1     20 4B 45
 1FF4     59 20 54
 1FF7     4F 20 43
 1FFA     4F 4E 54
 1FFD     49 4E 55
 2000     45 20 2D
 2003     2D 2D

 2005     AD 1DB4        L2005:  LDA      IOBSLT
                        +        CMPBN    <#$60>,L2040
 200C     AD 1DB5                LDA      IOBDRV
                        +        CMPBN    <#$01>,L2040
 2013     20 1C10                JSR      PRNTBF
                        +        DMOVI    L1FC0,ACC
 201E     A2 1F                  LDX      #$1F
 2020     20 1E29                JSR      OUTMSG
 2023     20 1C10        L2023:  JSR      PRNTBF
                        +        DMOVI    L1FDF,ACC
 202E     A2 26                  LDX      #$26
 2030     20 1E29                JSR      OUTMSG
 2033     20 1B91                JSR      OUTBUF
 2036     20 FD0C                JSR      RDKEY
                        +        CMPBN    <#$8D>,L2023
 203D     20 1C10                JSR      PRNTBF
 2040                   +L2040:  MOV      <#$60>,IOBSLT
                        +        MOV      <#$01>,IOBDRV
 204A     60                     RTS

                                 PAGE
```

```


204B    20 1EBD     OPSVGM: JSR     L1EBD                   ; setup for disk I/O

204E    A2 00               LDX     #$00                    ; copy game release # to buffer
2050    A0 02               LDY     #HDRREL
                    +       MOV     <(FRZMEM),Y>,<<BUFFER,X>>
2057    E8                  INX
2058    C8                  INY
                    +       MOV     <(FRZMEM),Y>,<<BUFFER,X>>
205E    E8                  INX

                    +       DMOVI   PRGIDX,ACC              ; copy PC to buffer
2067    A0 03               LDY     #$03
2069    20 20DF             JSR     SVGMMV

                    +       DMOVI   LOCVAR,ACC              ; copy local variables to buffer
2074    A0 1E               LDY     #$1E
2076    20 20DF             JSR     SVGMMV

                    +       DMOVI   STKCNT,ACC              ; copy SP and SP save to buffer
2081    A0 06               LDY     #$06
2083    20 20DF             JSR     SVGMMV

2086    20 1E16             JSR     DWRBUF                  ; write it out
2089    B0 4E               BCS     SVGMFL                  ; fail if error

208B    A2 00               LDX     #$00                    ; copy lowest 256 bytes of stack
                    +       DMOVI   STKLIM,ACC              ; to buffer
2095    A0 00               LDY     #$00
2097    20 20DF             JSR     SVGMMV

209A    20 1E16             JSR     DWRBUF                  ; write it out
209D    B0 3A               BCS     SVGMFL                  ; fail if error

209F    A2 00               LDX     #$00                    ; copy high 192 bytes of stack
                    +       DMOVI   STKLIM+$0100,ACC        ; to buffer
20A9    A0 C0               LDY     #$C0
20AB    20 20DF             JSR     SVGMMV

20AE    20 1E16             JSR     DWRBUF                  ; write it out
20B1    B0 26               BCS     SVGMFL                  ; fail if error

                    +       DMOV    FRZMEM,ACC              ; figure out how many pages of
20BB    A0 0E               LDY     #HDRIMP                 ; impure storage there are to be
                    +       MOV     <(FRZMEM),Y>,ACD        ; written out, and set up for first
20C1    E6 E8               INC     ACD                     ; one

20C3    20 1E1E     L20C3:  JSR     DWRNXT                  ; write one page of impure storage
20C6    B0 11               BCS     SVGMFL                  ; fail if error
20C8    E6 E7               INC     ACC+1                   ; increment buffer address
                    +       DECBN   ACD,L20C3               ; decrement page count, loop if more

20CE    20 1E1E             JSR     DWRNXT                  ; write final page
20D1    B0 06               BCS     SVGMFL                  ; fail if error
```

```
    20D3    20 2005              JSR    L2005                    ; make sure we have game disk
    20D6    4C 0B84              JMP    PREDTR                   ; return true (no error)

    20D9    20 2005      SVGMFL: JSR    L2005                    ; make sure we have game disk
    20DC    4C 0B8D              JMP    PREDFL                   ; return false (error)


    20DF    88           SVGMMV: DEY                             ; copy memory into buffer to write
                        +        MOV    <(ACC),Y>,<<BUFFER,X>>
    20E5    E8                   INX
                        +        CPYBN  <#$00>,SVGMMV            ; if more, loop
    20EA    60                   RTS                             ; no, return

                                 PAGE
```

```

20EB    20 1EBD      OPRSGM: JSR    L1EBD              ; setup for disk I/O

20EE    20 1DFA              JSR    DRDBUF             ; read in a bufferful
                     +       JCS    RSGMFL             ; fail if error

20F6    A2 00                LDX    #$00               ; check release of game, fail if wrong
20F8    A0 02                LDY    #HDRREL
20FA    B1 BA                LDA    (FRZMEM),Y
                     +       CMPBN  <BUFFER,X>,L210A
2101    E8                   INX
2102    C8                   INY
2103    B1 BA                LDA    (FRZMEM),Y
                     +       CMPBE  <BUFFER,X>,L210D
210A    4C 218E      L210A:  JMP    RSGMFL

210D    A0 11        L210D:  LDY    #HDRFLG+1          ; preserve SCRIPT flag
                     +       MOV    <(FRZMEM),Y>,MDFLAG

2113    E8                   INX                       ; restore PC
                     +       DMOVI  PRGIDX,ACC
211C    A0 03                LDY    #$03
211E    20 2194              JSR    RSGMMV
                     +       MOV    <#$00>,PRGUPD

                     +       DMOVI  LOCVAR,ACC         ; restore local variables
212D    A0 1E                LDY    #$1E
212F    20 2194              JSR    RSGMMV

                     +       DMOVI  STKCNT,ACC         ; restore SP and SP save
213A    A0 06                LDY    #$06
213C    20 2194              JSR    RSGMMV

213F    20 1DFA              JSR    DRDBUF             ; read a bufferful
2142    B0 4A                BCS    RSGMFL             ; fail if error

2144    A2 00                LDX    #$00               ; restore first 256 bytes of stack
                     +       DMOVI  STKLIM,ACC
214E    A0 00                LDY    #$00
2150    20 2194              JSR    RSGMMV

2153    20 1DFA              JSR    DRDBUF             ; read a bufferful
2156    B0 36                BCS    RSGMFL             ; fail if error

2158    A2 00                LDX    #$00               ; restore last 192 bytes of stack
                     +       DMOVI  STKLIM+$0100,ACC
2162    A0 C0                LDY    #$C0
2164    20 2194              JSR    RSGMMV

                     +       DMOV   FRZMEM,ACC         ; figure out how many pages of
216F    A0 0E                LDY    #HDRIMP            ; impure storage there are to be
                     +       MOV    <(FRZMEM),Y>,ACD   ; read in, and set up to read first
2175    E6 E8                INC    ACD                ; one

```

```
2177    20 1E02    L2177:  JSR     DRDNXT              ; read in next page of impure storage
217A    B0 12              BCS     RSGMFL              ; fail if error
217C    E6 E7              INC     ACC+1               ; increment buffer pointer
                   +       DECBN   ACD,L2177           ; decrement page count, loop if more

2182    A5 EA              LDA     MDFLAG              ; restore SCRIPT flag
2184    A0 11              LDY     #HDRFLG+1
2186    91 BA              STA     (FRZMEM),Y

2188    20 2005            JSR     L2005               ; make sure we have game disk
218B    4C 0B84            JMP     PREDTR              ; return true (no error)

218E    20 2005    RSGMFL: JSR     L2005               ; make sure we have game disk
2191    4C 0B8D            JMP     PREDFL              ; return false (error)


2194    88         RSGMMV: DEY                         ; copy buffer to memory (read)
                   +       MOV     <BUFFER,X>,<<(ACC),Y>>
219A    E8                 INX
                   +       CPYBN   <#$00>,RSGMMV
219F    60                 RTS


                           PAGE
```

```
 21A0    E6 4E        L21A0:  INC     RNDLOC               ; get a 'random' number
 21A2    E6 4F                INC     RNDLOC+1
                     +        DMOV    RNDLOC,ACC
 21AC    60                   RTS

 21AD    2D 2D 20     ENDMSG: DB      '-- END OF SESSION --''
 21B0    45 4E 44
 21B3    20 4F 46
 21B6    20 53 45
 21B9    53 53 49
 21BC    4F 4E 20
 21BF    2D 2D
 0014                 ENMSLN  EQU     *-ENDMSG

 21C1    49 4E 54     FTLMSG: DB      'INTERNAL ERROR #'
 21C4    45 52 4E
 21C7    41 4C 20
 21CA    45 52 52
 21CD    4F 52 20
 21D0    23
 0010                 FTMSLN  EQU     *-FTLMSG

 21D1    20 1C10      FATAL:  JSR     PRNTBF               ; flush anything left in buffer

                     +        DMOVI   FTLMSG,ACC           ; output fatal message
 21DC    A2 10                LDX     #FTMSLN
 21DE    20 1E29              JSR     OUTMSG

                     +        DPUL2   ACC                  ; output address where error detected
 21E7    20 14F5              JSR     PRNTNM

 21EA    20 1C10      OPENDS: JSR     PRNTBF               ; flush anything left in buffer

                     +        DMOVI   ENDMSG,ACC           ; output end of session message
 21F5    A2 14                LDX     #ENMSLN
 21F7    20 1E29              JSR     OUTMSG

 21FA    20 1C10              JSR     PRNTBF               ; flush the buffer

 21FD    4C 21FD      HALT:   JMP     HALT                 ; die horribly

                              .DEPHASE

                              END     START
```

```
Macros:
ADD             CMPBE           CMPBG           CMPBL           CMPBM
CMPBN           CMPBP           CMPJE           CMPJL           CMPJSE
CMPJSG          CMPJSN          CMPRE           CPXBE           CPXBG
CPXRGT          CPYBN           D1COMP          DADC            DADD
DADDB1          DADDB2          DAND            DASL            DDEC
DDEC2           DECA            DECABE          DECABM          DECABN
DECABP          DECBE           DECBN           DECJE           DECJN
DINC            DLSR            DMOV            DMOVI           DMOVI2
DOR             DPSH            DPUL            DPUL2           DROL
DROR            DSBC            DSTZ            DSUB            DSUBB1
DSUBB2          DTS2BE          DTS2BN          DTS2JE          DTS2JN
DTS2RE          DTS2RN          DTST            DTST2           DTSTBE
DTSTBN          DTSTJE          DTSTJN          DTSTRE          DTSTRN
DXBEQ           DXBMI           DXBNE           DXBPL           DYBEQ
DYBMI           DYBNE           DYBPL           INCA            IXBNE
IYBNE           JCC             JCS             JEQ             JGE
JGT             JLT             JMI             JNE             JPL
JSRCC           JSRCS           JSREQ           JSRGE           JSRGT
JSRLT           JSRMI           JSRNE           JSRPL           MOV
PSH             PUL             RTSCC           RTSCS           RTSEQ
RTSGE           RTSGT           RTSLT           RTSMI           RTSNE
RTSPL           STR             SUB             TSTA            TSTABE
TSTABM          TSTABN          TSTABP          TSTAJE          TSTARP

Symbols:
099F    ..0000          09A6    ..0001          09BE    ..0002
09C8    ..0003          09D2    ..0004          09FE    ..0005
0A38    ..0006          0A42    ..0007          0A70    ..0008
0A87    ..0009          0B4D    ..000A          0B9B    ..000B
0BD4    ..000C          0BE5    ..000D          0BFC    ..000E
0CB1    ..000F          0CDA    ..0010          0CE6    ..0011
0D38    ..0012          0D4E    ..0013          0D70    ..0014
0D8D    ..0015          0E00    ..0016          0E2F    ..0017
0E48    ..0018          0E8F    ..0019          0F10    ..001A
101A    ..001B          104A    ..001C          1078    ..001D
1084    ..001E          10B1    ..001F          12B8    ..0020
12CD    ..0021          131A    ..0022          1321    ..0023
132A    ..0024          13BF    ..0025          13C9    ..0026
13E0    ..0027          145B    ..0028          1466    ..0029
147B    ..002A          1486    ..002B          14BF    ..002C
14CA    ..002D          14FC    ..002E          1552    ..002F
1611    ..0030          1624    ..0031          163D    ..0032
1643    ..0033          16FF    ..0034          1710    ..0035
171F    ..0036          172C    ..0037          1736    ..0038
173D    ..0039          174C    ..003A          1756    ..003B
17F6    ..003C          1800    ..003D          18C4    ..003E
1A43    ..003F          1A5F    ..0040          1B48    ..0041
1B82    ..0042          1B9C    ..0043          1E08    ..0044
1E15    ..0045          1E24    ..0046          1F45    ..0047
1F63    ..0048          1F94    ..0049          20F6    ..004A
00E4    ACB             00E6    ACC             00E8    ACD
169D    ADVPPT          0082    ARG1            0084    ARG2
0086    ARG3            0088    ARG4            0081    ARGCNT
0093    AUXIDX          0091    AUXLPG          0094    AUXMPT
```

```
0097    AUXPPG          0096    AUXUPD          1B3F    BFCHAR
0200    BUFFER          00EC    CHRPT2          00EB    CHRPTR
FC9C    CLREOL          1B16    CLRSCR          FDED    COUT
FDF0    COUT1           000D    CRCHAR          1A05    CRNWRD
0036    CSWL            0024    CURSRH          0025    CURSRV
1DC4    DCT             1DC8    DISKIO          15AD    DIVIDE
18FC    DOASCI          1915    DOCRLF          18BE    DONEXT
193B    DOSBWD          1910    DOSPAC          18EB    DOSPCL
1E0D    DRDBKF          1E08    DRDBLK          1DFA    DRDBUF
1E02    DRDNXT          1BF5    DSPBUF          0A11    DSPTCH
1E16    DWRBUF          1E1E    DWRNXT          21AD    ENDMSG
0014    ENMSLN          21D1    FATAL           000C    FFCHAR
2C00    FIRFLC          1B1E    FNDMEM          1897    FNDPAG
00BA    FRZMEM          00BC    FRZPGS          17E8    FTAXBA
17DB    FTAXWD          21C1    FTLMSG          0010    FTMSLN
173E    FTPRBA          0AAB    FTPRBY          0AB5    FTPRWD
1D65    GETLIN          FD6F    GETLN1          19B9    GETNYB
0098    GLBVAR          0A04    GODOIT          1693    GTPLEN
168E    GTPNUM          0AEF    GTVARA          0AE8    GTVARP
1406    GTVCBA          0AC2    GTVRA1          21FD    HALT
001A    HDRCKA          001C    HDRCKV          0010    HDRFLG
0004    HDRFRZ          000C    HDRGBV          000E    HDRIMP
0000    HDRIRL          0002    HDRREL          0018    HDRSBW
0012    HDRSER          0006    HDRSTR          000A    HDRTHG
0001    HDRTYP          0008    HDRVCB          FC58    HOME
1AF7    INITSC          0032    INVFLG          00D3    INWORD
1DB3    IOB             1DBB    IOBBUF          1DBF    IOBCMD
1DB5    IOBDRV          1DB8    IOBSCT          1DB4    IOBSLT
1DB7    IOBTRK          0805    L0805           084A    L084A
0897    L0897           08B6    L08B6           090A    L090A
09AF    L09AF           09D7    L09D7           09ED    L09ED
0A2B    L0A2B           0A45    L0A45           0A73    L0A73
0A8A    L0A8A           0A98    L0A98           0AD0    L0AD0
0AD6    L0AD6           0B02    L0B02           0B26    L0B26
0B60    L0B60           0B94    L0B94           0B9C    L0B9C
0BAD    L0BAD           0BC3    L0BC3           0BDA    L0BDA
0BFC    L0BFC           0C17    L0C17           0C1A    L0C1A
0CA5    L0CA5           0CDA    L0CDA           0CE6    L0CE6
0CFA    L0CFA           0D4E    L0D4E           0DB7    L0DB7
0DD2    L0DD2           0DE4    L0DE4           0E2F    L0E2F
0E4C    L0E4C           0E9D    L0E9D           0EB7    L0EB7
0ECF    L0ECF           0F08    L0F08           0F10    L0F10
0F23    L0F23           0F97    L0F97           0FA1    L0FA1
0FD1    L0FD1           100B    L100B           103B    L103B
105E    L105E           106C    L106C           107E    L107E
108E    L108E           10A5    L10A5           10B7    L10B7
10BD    L10BD           1104    L1104           1107    L1107
110D    L110D           1111    L1111           1139    L1139
113F    L113F           1143    L1143           1173    L1173
117F    L117F           118E    L118E           119D    L119D
11A0    L11A0           11B4    L11B4           11F2    L11F2
1220    L1220           1230    L1230           124C    L124C
12AC    L12AC           12BE    L12BE           12D7    L12D7
1310    L1310           1332    L1332           135C    L135C
137A    L137A           1382    L1382           13BA    L13BA
13DA    L13DA           13E0    L13E0           13E4    L13E4
```

```
13ED    L13ED           13EF    L13EF           13F0    L13F0
13F1    L13F1           13FB    L13FB           141F    L141F
1445    L1445           144A    L144A           1450    L1450
1470    L1470           148E    L148E           14B4    L14B4
14D0    L14D0           14D7    L14D7           1500    L1500
1519    L1519           151D    L151D           1529    L1529
152E    L152E           1568    L1568           1578    L1578
158B    L158B           15C5    L15C5           15D6    L15D6
15FB    L15FB           160A    L160A           1611    L1611
161F    L161F           1643    L1643           1653    L1653
165D    L165D           16A1    L16A1           16C3    L16C3
16CB    L16CB           16DE    L16DE           16E9    L16E9
16F3    L16F3           1757    L1757           1761    L1761
1770    L1770           1778    L1778           1788    L1788
1790    L1790           17C4    L17C4           1801    L1801
1807    L1807           180B    L180B           181A    L181A
1822    L1822           1832    L1832           183A    L183A
1891    L1891           1892    L1892           189D    L189D
18A9    L18A9           18B1    L18B1           18D9    L18D9
18DC    L18DC           18E2    L18E2           192D    L192D
1937    L1937           193A    L193A           19B4    L19B4
19BF    L19BF           19D6    L19D6           19F3    L19F3
19FF    L19FF           1A09    L1A09           1A1B    L1A1B
1A2A    L1A2A           1A43    L1A43           1A52    L1A52
1A62    L1A62           1A6C    L1A6C           1A99    L1A99
1A9B    L1A9B           1AA6    L1AA6           1AB6    L1AB6
1AC1    L1AC1           1AC9    L1AC9           1ACA    L1ACA
1B28    L1B28           1B57    L1B57           1B61    L1B61
1B64    L1B64           1B66    L1B66           1B70    L1B70
1B77    L1B77           1BA0    L1BA0           1BD5    L1BD5
1BE3    L1BE3           1BF7    L1BF7           1C05    L1C05
1C40    L1C40           1C50    L1C50           1C79    L1C79
1C89    L1C89           1CB8    L1CB8           1CDB    L1CDB
1CF5    L1CF5           1D00    L1D00           1D05    L1D05
1D1C    L1D1C           1D33    L1D33           1D35    L1D35
1D40    L1D40           1D43    L1D43           1D59    L1D59
1D8B    L1D8B           1D94    L1D94           1D98    L1D98
1DA7    L1DA7           1DB1    L1DB1           1DDF    L1DDF
1DE7    L1DE7           1DED    L1DED           1E2F    L1E2F
1E3D    L1E3D           1E59    L1E59           1E5A    L1E5A
1E69    L1E69           1E6C    L1E6C           1E7B    L1E7B
1E7E    L1E7E           1E8D    L1E8D           1E90    L1E90
1E9A    L1E9A           1EBD    L1EBD           1F12    L1F12
1F3A    L1F3A           1F48    L1F48           1F4C    L1F4C
1FB3    L1FB3           1FC0    L1FC0           1FDF    L1FDF
2005    L2005           2023    L2023           2040    L2040
20C3    L20C3           210A    L210A           210D    L210D
2177    L2177           21A0    L21A0           0001    LC40
00D9    LD9             00DE    LDE             00DF    LDF
0100    LDORG           00E0    LE0             00E1    LE1
000A    LFCHAR          00ED    LINCNT          009A    LOCVAR
00BF    LRUPAG          BFFF    LSTFLC          0800    MAINOR
00EA    MDFLAG          098F    MNLOOP          1C0A    MOREMS
1862    MRKPAG          0006    MRMSLN          00BE    MRUPAG
191F    NEWMOD          10C3    OPADD           0F4A    OPAND
11A3    OPCALL          0A66    OPCGPA          0A2E    OPCGPB
```

```
0A17    OPCGPC          09AA    OPCGPD          0C7C    OPCKSM
0F80    OPCLRA          0080    OPCODE          0C72    OPCRLF
0D60    OPDEC           0EE7    OPDECB          1118    OPDIV
0D81    OPDSTT          21EA    OPENDS          0FEC    OPGTBY
0CF3    OPGTCH          12DC    OPGTLN          109E    OPGTNP
1008    OPGTP           1069    OPGTPA          0D20    OPGTPL
0D0E    OPGTPR          0CE9    OPGTSB          0FD2    OPGTWD
0D43    OPINC           0EF5    OPINCB          0E7C    OPJUMP
000E    OPMAX1          0010    OPMAX2          0019    OPMAX3
000A    OPMAX4          0EA0    OPMOVE          0FA4    OPMOVT
116B    OPMTCH          10E3    OPMUL           0EA8    OPNOT
0C53    OPNULL          0F3B    OPOR            14E5    OPPRCH
14EA    OPPRNM          1C8A    OPPRST          0DE2    OPPRTN
0D73    OPPSB           0C28    OPPSI           0C54    OPPSIC
0E92    OPPSW           1288    OPPTBY          12A9    OPPTP
125F    OPPTWD          1560    OPPULL          1555    OPPUSH
114A    OPRMD           1536    OPRNDM          20EB    OPRSGM
0E06    OPRTN           0C23    OPRTNF          0C18    OPRTNT
0C64    OPRTNV          0F6D    OPSETA          10D3    OPSUB
204B    OPSVGM          090D    OPTAB1          0929    OPTAB2
0949    OPTAB3          097B    OPTAB4          0F13    OPTINT
0F59    OPTSTA          0CDD    OPTSTZ          1B91    OUTBUF
1E29    OUTMSG          00DA    PKWORD          00D1    PNYBBF
00D0    PNYBCN          00EE    PRCSWL          0B8D    PREDFL
0B84    PREDTR          008A    PRGIDX          008B    PRGLPG
008D    PRGMPT          0090    PRGPPG          008F    PRGUPD
00CF    PRMMOD          1C10    PRNTBF          14F5    PRNTNM
18B4    PRNTST          0033    PROMPT          1BA1    PRTBUF
0779    PRTWDT          0B46    PTVARA          0B35    PTVARP
0AC9    PTVRA1          0B32    PTVRP1          0B2C    PTVRPA
0B2A    PTVRPZ          1720    PULLWD          16F4    PUSHWD
FD0C    RDKEY           004E    RNDLOC          0000    RNGDBG
218E    RSGMFL          2194    RSGMMV          2900    RWTS
2400    RWTSOR          00E2    SBWDPT          0006    SCMSLN
1C7E    SCORMS          007F    SECPTK          13D2    SEPTAB
17B8    SETAXB          17C9    SETAXW          1629    SETUPA
1669    SETUPP          16A7    SETUPT          1D57    SHWMSG
1995    SPCLCH          0800    START           03E8    STCKLC
00E0    STCKMX          00C8    STKCNT          00CD    STKCSV
0228    STKLIM          00C9    STKPNT          00CB    STKPSV
00F3    STLTYP          20D9    SVGMFL          20DF    SVGMMV
00B8    SWPMEM          00BD    SWPPGS          0009    TBCHAR
0000    THGATT          0006    THGCHD          0004    THGPAR
0007    THGPRP          0005    THGSIB          1C84    TIMEMS
0005    TMMSLN          00CE    TMPMOD          1AAB    TSTCHR
19AD    TSTMOD          0000    VERSN           2200    VMT1LC
2280    VMT2LC          2300    VMT3LC          2380    VMT4LC
00C0    VMTAB1          00C2    VMTAB2          00C4    VMTAB3
00C6    VMTAB4          2200    VMTORG          FC22    VTAB
0023    WNDBOT          0020    WNDLFT          0022    WNDTOP
0021    WNDWDT          007F    ZPORG


No Fatal error(s)

C
```

```
..0000   1228   1228#
..0001   1229   1229#
..0002   1249   1249#
..0003   1250   1250#
..0004   1251   1251#
..0005   1277   1277#
..0006   1307   1307#
..0007   1308   1308#
..0008   1326   1326#
..0009   1332   1332#
..000A   1422   1422#
..000B   1456   1456#
..000C   1481   1481#
..000D   1483   1483#
..000E   1493   1493#
..000F   1575   1575#
..0010   1591   1591#
..0011   1597   1597#
..0012   1624   1624#
..0013   1634   1634#
..0014   1646   1646#
..0015   1664   1664#
..0016   1704   1704#
..0017   1722   1722#
..0018   1729   1729#
..0019   1752   1752#
..001A   1789   1789#
..001B   1903   1903#
..001C   1924   1924#
..001D   1941   1941#
..001E   1944   1944#
..001F   1963   1963#
..0020   2193   2193#
..0021   2203   2203#
..0022   2229   2229#
..0023   2230   2230#
..0024   2232   2232#
..0025   2299   2299#
..0026   2303   2303#
..0027   2314   2314#
..0028   2369   2369#
..0029   2370   2370#
..002A   2376   2376#
..002B   2377   2377#
..002C   2400   2400#
..002D   2401   2401#
..002E   2426   2426#
..002F   2460   2460#
```

```
..0030    2530    2530#
..0031    2540    2540#
..0032    2553    2553#
..0033    2554    2554#
..0034    2653    2653#
..0035    2656    2656#
..0036    2660    2660#
..0037    2665    2665#
..0038    2667    2667#
..0039    2669    2669#
..003A    2684    2684#
..003B    2688    2688#
..003C    2782    2782#
..003D    2786    2786#
..003E    2887    2887#
..003F    3058    3058#
..0040    3066    3066#
..0041    3185    3185#
..0042    3223    3223    3223#
..0043    3240    3240#
..0044    3512    3512#
..0045    3517    3517#
..0046    3521    3521#
..0047    3634    3634#
..0048    3643    3643#
..0049    3653    3653#
..004A    3794    3794#
ACB        174#   1070    1070    1087    1089    1395    1398    1400    1400    1400    1400    1402
          1404    1433    1436    1438    1438    1438    1438    1440    1442    1485    1488    1493
          1493    1496    1496    1498    1501    1672    1672    1673    1673    1674    1683    1683
          1684    1684    1685    1699    1701    1702    1702    1719    1719    1722    1722    1722
          1726    1728    1729    1729    1729    1770    1770    1775    1775    1782    1782    1786
          1786    1800    1803    1819    1821    1822    1824    1833    1837    1849    1854    1907
          1911    1917    1919    1925    1927    1928    1928    1988    1988    1989    1991    2007
          2007    2008    2010    2024    2024    2027    2027    2115    2116    2121    2126    2127
          2285    2287    2356    2358    2370    2370    2370    2370    2373    2377    2377    2377
          2401    2401    2401    2402    2402    2403    2403    2405    2405    2429    2429    2431
          2456    2456    2459    2459    2480    2487    2487    2489    2489    2490    2490    2503
          2506    2516    2516    2523    2554    2557    2564    2566    2575    2577    2578    2578
          2640    2643    2648    2649    2725    2725    2823    2823    3044    3054    3056    3062
          3064    3074    3076    3083    3085    3496    3497    3512    3512    3521    3521    3629
          3629    3634    3634    3634
ACC        175#   1069    1069    1082    1084    1260    1261    1275    1275    1284    1286    1296
          1296    1312    1312    1318    1318    1328    1328    1334    1334    1342    1342    1350
          1352    1360    1362    1376    1376    1378    1378    1389    1391    1402    1404    1412
          1414    1417    1417    1420    1420    1427    1429    1440    1442    1463    1465    1471
          1473    1475    1477    1480    1480    1481    1481    1481    1482    1482    1483    1483
          1483    1485    1491    1547    1547    1563    1571    1608    1608    1610    1619    1619
          1620    1623    1623    1624    1624    1624    1634    1634    1635    1635    1638    1638
```

```
         1646    1646    1646    1652    1652    1662    1666    1666    1669    1676    1680    1685
         1686    1686    1688    1689    1699    1701    1702    1702    1704    1704    1715    1720
         1726    1728    1733    1733    1736    1736    1739    1740    1744    1744    1751    1751
         1752    1752    1752    1756    1756    1766    1766    1769    1769    1776    1776    1786
         1786    1787    1787    1796    1799    1802    1809    1809    1812    1812    1835    1839
         1850    1855    1861    1861    1868    1868    1870    1872    1875    1876    1876    1879
         1884    1884    1889    1889    1891    1893    1917    1919    1925    1927    1928    1928
         1930    1930    1931    1944    1944    1946    1947    1949    1951    1951    1951    1951
         1975    1975    1981    1981    1987    1987    1998    1998    1999    1999    2006    2006
         2016    2016    2017    2017    2023    2023    2027    2027    2060    2060    2064    2065
         2068    2068    2071    2071    2090    2092    2099    2101    2104    2106    2116    2118
         2120    2123    2125    2132    2150    2153    2156    2156    2156    2156    2159    2161
         2171    2174    2177    2177    2177    2177    2180    2205    2208    2211    2327    2333
         2338    2340    2341    2341    2341    2341    2345    2348    2356    2358    2361    2361
         2362    2366    2369    2369    2369    2376    2376    2376    2386    2390    2394    2398
         2400    2400    2400    2405    2405    2419    2419    2424    2428    2428    2459    2459
         2460    2460    2467    2467    2485    2485    2489    2489    2499    2499    2500    2500
         2502    2505    2508    2510    2513    2513    2516    2516    2517    2517    2521    2532
         2533    2535    2536    2553    2553    2554    2554    2564    2566    2575    2577    2578
         2578    2579    2588    2595    2615    2617    2617    2621    2621    2621    2621    2621
         2621    2622    2623    2626    2628    2631    2631    2633    2635    2641    2644    2646
         2648    2655    2657    2664    2666    2696    2696    2723    2723    2724    2724    2740
         2741    2748    2751    2765    2767    2794    2794    2821    2821    2822    2822    2843
         2846    2847    2849    2852    2856    2858    2865    2867    2869    2957    2959    3000
         3000    3044    3044    3045    3166    3166    3167    3168    3170    3171    3173    3174
         3175    3311    3311    3359    3370    3370    3380    3380    3385    3392    3400    3409
         3428    3495    3495    3511    3511    3520    3520    3531    3595    3595    3623    3623
         3642    3642    3643    3643    3643    3649    3649    3652    3652    3653    3653    3653
         3703    3703    3707    3707    3731    3731    3735    3735    3739    3739    3747    3747
         3755    3755    3762    3762    3768    3783    3810    3810    3815    3815    3819    3819
         3827    3827    3835    3835    3839    3839    3845    3861    3871    3871    3893    3893
         3897    3897    3902    3902
ACD       176#   1720    1730    1820    1823    1834    1838    1847    1852    2112    2113    2130
         2219    2221    2230    2230    2230    2232    2254    2256    2264    2266    2267    2271
         2272    2274    2277    2281    2294    2295    2297    2304    2305    2353    2369    2376
         2377    2382    2400    2427    2431    2434    2440    2478    2478    2479    2479    2485
         2485    2485    2485    2486    2486    2490    2490    2491    2491    2498    2498    2499
         2499    2512    2512    2517    2517    2518    2518    2556    2556    2560    2560    2887
         2895    2902    2915    2917    3043    3047    3050    3058    3058    3062    3066    3069
         3070    3078    3078    3087    3087    3527    3529    3530    3533    3535    3764    3764
         3770    3841    3841    3847
ADD       357#   1055    1079    1107    1112    1115    1490    1575    1626    1722    2292    2360
         2437    2621    2630    2710    2723    2808    2821    2894    3053    3097    3505    3634
         3643    3653
ADVPPT   1903    1941    1963    1967    2194    2605#
ARG1      118#   1260    1261    1312    1312    1328    1328    1513    1515    1547    1547    1563
         1563    1575    1575    1575    1588    1591    1597    1597    1599    1604    1615    1623
         1623    1631    1635    1643    1652    1652    1659    1671    1682    1694    1744    1744
         1751    1751    1756    1756    1760    1766    1766    1769    1769    1775    1775    1791
```

```
          1799   1802   1809   1809   1812   1812   1861   1865   1875   1884   1884   1889
          1889   1975   1975   1981   1981   1987   1987   2006   2006   2023   2023   2036
          2038   2041   2043   2046   2048   2059   2059   2076   2076   2149   2152   2170
          2173   2217   2217   2217   2217   2246   2250   2259   2263   2300   2412   2419
          2419   2456   2456   2467   2467   2473   2547   2571   3456   3456   3466
ARG2      119#   1334   1334   1559   1561   1566   1567   1578   1580   1770   1770   1776
          1776   1782   1782   1787   1787   1794   1799   1802   1809   1809   1812   1812
          1861   1861   1870   1883   1883   1884   1884   1889   1889   1902   1912   1940
          1958   1962   1975   1975   1981   1981   1988   1988   2007   2007   2024   2024
          2038   2040   2117   2122   2145   2147   2151   2168   2172   2191   2218   2218
          2218   2218   2223   2226   2229   2244   2281   2285   2287   2293   2293   2549
ARG3      120#   1563   1568   1582   2043   2045   2159   2161   2180   2205   2208   2211
ARG4      121#   1563   1565   2048   2050
ARGCNT    116#   1223   1264   1311   1336   2033   2112
AUXIDX    130#   1522   1528   1576   2740   2750   2776   2780   2962   2967
AUXLPG    129#   1523   1523   1529   1529   1578   1580   2741   2742   2753   2756   2786
          2786   2788   2790   2794   2794   2823   2823   2828   2829   2963   2963   2966
          2966
AUXMPT    131#   1531   1531   2777   2809   2810
AUXPPG    133#   2719   2795   2803   2822   2826
AUXUPD    132#   1040   1524   1530   2720   2743   2773   2784   2812   2968
BFCHAR    1538   1540   1551   1553   2413   2438   2443   2446   2895   2926   3182#  3395
          3403   3406   3414   3416   3532   3601   3612   3620
BUFFER     43#   3197   3209   3224   3226   3272   3291   3442   3460   3511   3511   3520
          3520   3725   3728   3783   3799   3803   3861
CHRPT2    181#   3213   3220   3221   3229
CHRPTR    180#   3182   3201   3214   3225   3227   3271   3290   3298   3321   3450   3453
CLREOL     97#   3317   3364   3421   3659
CLRSCR    3158#  3591
CMPBE     939#   1085   1795   1901   1922   1939   1961   1992   1993   2011   2012   2039
          2044   2049   2190   2201   2316   2332   2838   2853   2868   2937   3092   3208
          3323   3360   3467   3802
CMPBG     954#   1279   1293   1315   1339   1384   1422   2367   3101   3105   3401   3463
          3666
CMPBL     949#   1229   2385   2389   2393   2397   2550   2693   2791   2889   2890   3100
          3104   3185   3186   3309   3455   3462   3665
CMPBM     959#   3388
CMPBN     944#   1577   1579   1581   1587   1670   1681   2037   2042   2047   2647   2718
          2816   2866   2897   3169   3172   3627   3662   3698   3700   3711   3798
CMPBP     964#   3410
CMPJE     969#   1590   3184
CMPJL     974#   1227   1228   1276
CMPJSE    979#   1249   1250   1307   2231
CMPJSG    989#   2659
CMPJSN    984#   1923   2202
CMPRE     994#   2228
COUT      100#   3260   3264   3266   3268   3273   3333
COUT1     101#   3292   3325   3430
CPXBE     999#   3270   3289
```

```
CPXBG     1004#   3198
CPXRGT    1009#   3222
CPYBN     1014#   3784    3862
CRCHAR      68#   1537    1550    2309    2925    3185    3468
CRNWRD    2281    3037#
CSWL        88#   3249    3249    3252    3252    3279    3279    3282    3282    3331    3331    3332
          3332    3335    3335    3336    3336
CURSRH      81#   3250    3281    3317    3317    3353    3354    3367    3372    3382    3424    3647
          3659
CURSRV      82#   3353    3354    3424
D1COMP     312#   1765
DADC       321#   1400    1438    1623    1884    1889    1975    2156    2177    2217    2218    2341
          2484    2578
DADD       347#   1399    1437    1622    1883    1888    1974    2155    2176    2216    2217    2340
          2577
DADDB1     383#   2368    2376    2399
DADDB2     405#   1574    1721    3633    3642    3652
DAND       303#   1811
DASL       238#   1882    1997    1998    2075    2559    2621    2621    2621
DCT       3481    3490#
DDEC       437#   1480    1482    1623    1645    1751    2400    2652    2655
DDEC2      441#   1728
DECA       518#   1423    2946    3007    3061
DECABE     889#
DECABM     904#
DECABN     894#   3006
DECABP     899#   3060
DECBE      869#   2112    3077    3086
DECBN      874#   1729    2129    2439    3534    3769    3846
DECJE      879#   3057
DECJN      884#   3065
DINC       429#   1492    1633    1703    1943    2459    2552    2553    2664    2666    2687    2785
          3511    3520
DISKIO    3493#   3513    3522
DIVIDE    2013    2025    2429    2457    2497#
DLSR       252#   2015    2016
DMOV       445#   1068    1311    1327    1333    1522    1528    1530    1546    1651    1701    1711
          1732    1735    1743    1750    1755    1768    1769    1774    1775    1781    1785    1786
          1860    1927    1986    1987    2005    2006    2022    2023    2026    2067    2070    2136
          2418    2455    2458    2466    2488    2489    2498    2516    2695    2722    2724    2793
          2820    2822    2999    3251    3278    3331    3334    3494    3761    3838    3870
DMOVI      456#   1042    1046    1047    1048    1049    1066    1069    1274    1295    1317    1341
          1718    2059    2428    2478    2499    2555    3310    3369    3379    3510    3519    3594
          3622    3641    3648    3651    3702    3706    3730    3734    3738    3746    3754    3809
          3814    3818    3826    3834    3892    3901
DMOVI2     467#   3165
DOASCI    2902    2909#
DOCRLF    2903    2925#
DONEXT    2885#   2896    2935    2940    2971
```

```
DOR       294#    1808
DOSBWD   2890     2946#
DOSPAC   2888     2922#
DOSPCL   2898     2901#
DPSH      502#    1375     1416     1634     1665     1672     1683     1867     2477     2497     2960     2962
         3248     3330
DPUL      492#    1377     1419     1637     1671     1682     1685     1875     2490     2517     2965     2968
         3281     3335
DPUL2     497#    3896
DRDBKF   1070     1089     2726     2824     3515#
DRDBLK   3512#    3515
DRDBUF   3510#    3792     3822     3830
DRDNXT   3511#    3843
DROL      280#    2511     2512
DROR      266#    2485     2486     2515
DSBC      334#    1951     1981     2405
DSPBUF   3240     3287#    3363     3420
DSPTCH   1284     1286     1286#    1286
DSTZ      232#
DSUB      352#    1950     1980     2404
DSUBB1    394#    2369     2375
DSUBB2    417#    1481     1483     1624     1646     1729     1752     2401     2653     2656
DTS2BE    789#
DTS2BN    794#    2058     2401
DTS2JE    799#
DTS2JN    804#
DTS2RE    809#
DTS2RN    814#
DTST      749#    1480     1482     1496     1597     2230     2428
DTST2     784#    2059     2402
DTSTBE    754#    1479     1481     1495     2427
DTSTBN    759#
DTSTJE    764#
DTSTJN    769#    1596
DTSTRE    774#    2229
DTSTRN    779#
DWRBUF   3519#    3742     3750     3758
DWRNXT   3520#    3766     3771
DXBEQ     829#    2040     2045
DXBMI     849#    3501
DXBNE     819#    1059     2034     2318     2487     2513     2871     3209     3432     3468
DXBPL     839#    2608     3093
DYBEQ     834#
DYBMI     854#
DYBNE     824#    2238     3041     3634
DYBPL     844#
ENDMSG   3873#    3880     3902     3902
ENMSLN   3880#    3902
FATAL    1129     1162     1177     1299     1924     2035     2193     2203     2660     2669     3517     3890#
```

```
FFCHAR     71#   2310
FIRFLC     54#   1067    1067
FNDMEM   1117    3165#
FNDPAG   2696    2794    2863#
FRZMEM    139#   1067    1067    1069    1069    1074    1076    1081    1084    1094    1100    1102
         1106    1108    1108    1111    1113    1113    1116    1559    1561    1586    1589    1623
         1623    1905    1906    1909    1910    1951    1951    2156    2156    2177    2177    2217
         2217    2218    2218    2338    2340    2341    2341    2405    2405    2578    2578    2631
         2632    2634    2711    2809    3237    3327    3446    3725    3728    3762    3762    3764
         3797    3801    3807    3839    3839    3841    3850
FRZPGS    140#   1076    1076    1086    1116    1584    2694    2706    2792    2804
FTAXBA   1573    1890    2762    2764    2773#   2812
FTAXWD   1885    2762#   2998
FTLMSG   3882#   3888    3893    3893
FTMSLN   3888#   3893
FTPRBA   1224    1235    1349    1357    1359    1382    1417    1446    1450    1456    1470    2080
         2096    2098    2675#   2714
FTPRBY   1251    1308    1326    1332    1349#
FTPRWD   1249    1307    1357#
GETLIN   2218    3437#
GETLN1     99#   3439
GETNYB   2885    2909    2914    2951    2992#
GLBVAR    135#   1106    1108    1400    1400    1438    1438
GODOIT   1281#   1297    1319    1343
GTPLEN   1625    1920    2199    2595#   2605
GTPNUM   1900    1938    1960    1965    2189    2588#
GTVARA   1366    1384#
GTVARP   1250    1308    1326    1332    1382#
GTVCBA   2325    2336#   2343
GTVRA1   1365#   1632    1644    1761    3358    3374    3384    3397    3408
HALT     3907#   3907
HDRCKA    210#   1557
HDRCKV    211#   1585
HDRFLG    207#   3236    3326    3445    3805    3849
HDRFRZ    201#   1072
HDRGBV    205#   1104
HDRIMP    206#   3762    3839
HDRIRL    198#
HDRREL    200#   3723    3796
HDRSBW    209#   1109
HDRSER    208#
HDRSTR    202#   1098
HDRTHG    204#   1903    2629
HDRTYP    199#   1093
HDRVCB    203#   2336
HOME       96#   3158
INCA      514#
INITSC   1037    3144#
INVFLG     84#   3155    3189    3313    3315    3356    3423    3648    3656
```

```
INWORD    161#   2265   2270   3047
IOB      3475#   3506   3507
IOBBUF   3482#   3495   3495
IOBCMD   3484#   3493
IOBDRV   3477#   3617   3699   3715
IOBSCT   3480#   3506
IOBSLT   3476#   3609   3697   3714
IOBTRK   3479#   3496   3503
IXBNE     859#   1032
IYBNE     864#
JCC       544#   1940   1962
JCS       551#   1788   3793
JEQ       530#   1422   1591   3058   3185
JGE       565#
JGT       572#
JLT       558#   1228   1229   1277
JMI       587#
JNE       537#   1597   3066
JPL       580#
JSRCC     614#   2192
JSRCS     624#   1902   3516
JSREQ     594#   1248   1250   1251   1306   1308   1325   1331   1455   2232   2668
JSRGE     644#   2660
JSRGT     654#
JSRLT     634#
JSRMI     675#   2425
JSRNE     604#   1924   2203   3239
JSRPL     665#
L0805    1031#   1033
L084A    1053#   1060
L0897    1079#   1091
L08B6    1086    1093#
L090A    1119    1129#
L09AF    1239#   1272
L09D7    1249    1250   1251   1256#
L09ED    1254    1274#
L0A2B    1280    1294   1299#  1316   1340
L0A45    1307    1308   1310#
L0A73    1326    1327#
L0A8A    1332    1333#
L0A98    1277    1338#
L0AD0    1366    1372#
L0AD6    1370    1375#
L0B02    1385    1393#
L0B26    1384    1406#
L0B60    1423    1431#
L0B94    1448    1454#
L0B9C    1448    1452   1458#
L0BAD    1460    1467#
```

```
L0BC3     1465      1474      1479#
L0BDA     1482#     1752
L0BFC     1491      1493#
L0C17     1496      1508#
L0C1A     1513#     1518
L0CA5     1573#     1578      1580      1582
L0CDA     1588      1592#
L0CE6     1597#     1613
L0CFA     1602      1607#
L0D4E     1634#     1646
L0DB7     1671      1678#     1682
L0DD2     1677      1685#
L0DE4     1695#     3362
L0E2F     1723#     1730
L0E4C     1716      1731#
L0E9D     1653      1757#
L0EB7     1179      1768#
L0ECF     1180      1774#
L0F08     1782      1787#
L0F10     1771      1777      1789#     1796      1805      1825
L0F23     1184      1798#
L0F97     1190      1860#
L0FA1     1862#     2474
L0FD1     1877      1880#
L100B     1900#     1903
L103B     1902      1920#
L105E     1923      1929#
L106C     1938#     1942
L107E     1940      1943#
L108E     1948      1950#
L10A5     1960#     1964
L10B7     1959      1965#     1968
L10BD     1962      1967#
L1104     1990      1994#
L1107     1995#     1999      2014      2017
L110D     1994      1997#
L1111     1993      1998#
L1139     2009      2013#
L113F     2013      2015#
L1143     2012      2016#
L1173     2035      2036#     2048
L117F     2038      2040#
L118E     2043      2045#
L119D     2041      2046      2050#
L11A0     2040      2045      2050      2051#
L11B4     2059      2063#
L11F2     2088#     2109
L1220     2084      2111#
L1230     2116#     2130
```

```
L124C     2113     2131#
L12AC     2189#    2195
L12BE     2191     2199#
L12D7     2202     2210#
L1310     2225#    2255     2269     2275     2295
L1332     2236#    2239
L135C     2233     2252     2256#
L137A     2248     2270#
L1382     2257     2261     2274#
L13BA     2232     2297#    2306
L13DA     2260     2301     2312#
L13E0     2251     2314#
L13E4     2316#    2319
L13ED     2319#    2330
L13EF     2317     2321#    2333
L13F0     2322#
L13F1     2247     2312     2324#
L13FB     2330#    2334
L141F     2282     2343#
L1445     2361     2363#
L144A     2366#    2372     2374
L1450     2364     2368#
L1470     2368     2371     2375#
L148E     2383#    2402
L14B4     2386     2390     2394     2399#
L14D0     2386     2390     2394     2398     2402#
L14D7     2398     2404#
L1500     2427#    2432
L1519     2428     2434#
L151D     2436#    2440
L1529     2435     2442#
L152E     2426     2445#
L1568     1994     2477#
L1578     2480#    2488
L158B     2483     2485#
L15C5     2501#    2514
L15D6     2507     2511#
L15FB     1988     2007     2024     2520#
L160A     1995     2527#
L1611     2447     2530#    2541
L161F     2522     2524     2539#
L1643     2551     2554#
L1653     2558#    2561
L165D     2558     2562#
L16A1     2607#    2609
L16C3     2622     2625#
L16CB     2627     2629#
L16DE     1770     1776     1787     2640#
L16E9     1804     2642     2646#
```

```
L16F3    2648    2650#
L1757    2676    2690#
L1761    2691    2695#
L1770    2702#   2733
L1778    2694    2710#
L1788    2698    2718#
L1790    2719    2722#
L17C4    2742#   2757
L1801    2774    2788#
L1807    1565    1584    2791#
L180B    2789    2793#
L181A    2800#   2831
L1822    2792    2808#
L1832    2796    2816#
L183A    2817    2820#
L1891    2839    2856#
L1892    2854    2857#
L189D    2866#   2872
L18A9    2867    2870#
L18B1    2869    2875#
L18D9    2894#   2899
L18DC    2895#   2907    2918    2923    2928
L18E2    2893    2897#
L192D    2933    2936#
L1937    2938    2940#
L193A    2944#   2950    2954
L19B4    2983    2986#
L19BF    2993    2996#
L19D6    2996    3006#
L19F3    3007    3024#
L19FF    3026    3028#
L1A09    3039#   3042
L1A1B    3044#   3066
L1A2A    3047    3050#
L1A43    3053    3058#
L1A52    3049    3062#   3069    3072    3089
L1A62    3061    3067#
L1A6C    3067    3070#
L1A99    3071    3091#
L1A9B    3092#   3094
L1AA6    3093    3096#
L1AB6    3101    3102    3104#
L1AC1    3105    3106    3108#
L1AC9    3109    3109    3111#
L1ACA    3058    3066    3078    3087    3113#
L1B28    3167#   3170    3173
L1B57    3187    3190    3195#
L1B61    3186    3201#   3337
L1B64    3199    3206#
```

```
L1B66     3208#     3210
L1B70     3209      3213#
L1B77     3220#     3230
L1BA0     3151      3246#     3255      3257
L1BD5     3256      3270#     3276
L1BE3     3271      3278#
L1BF7     3289#     3295
L1C05     3290      3297#
L1C40     3310      3320#
L1C50     3324      3326#
L1C79     3329      3336#
L1C89     3349#     3361      3361
L1CB8     3361      3366#
L1CDB     3368      3379#
L1CF5     3386      3388#
L1D00     3389      3389      3393#
L1D05     3377      3395#
L1D1C     3402      3404#
L1D33     3411      3413#
L1D35     3412      3414#
L1D40     3399      3419#
L1D43     3417      3420#
L1D59     3428#     3433
L1D8B     3448      3453#
L1D94     3456      3457#
L1D98     3460#     3469
L1DA7     3463      3464      3465#
L1DB1     3458      3468      3469#
L1DDF     3499#     3504
L1DE7     3500      3503#
L1DED     3502      3505#
L1E2F     3530#     3535
L1E3D     3537#     3595      3595
L1E59     3548#     3599      3603      3614      3643      3653      3661
L1E5A     3550#     3642      3642
L1E69     3555#     3611      3652      3652      3663      3666      3667
L1E6C     3557#
L1E7B     3562#     3619
L1E7E     3564#
L1E8D     3569#     3600      3629
L1E90     3571#     3649      3649
L1E9A     3576#     3623      3623
L1EBD     3591#     3720      3790
L1F12     3621#     3628
L1F3A     3633#     3635
L1F4B     3631      3635#
L1F4C     3599      3603      3614      3640#     3666      3667
L1FB3     3663      3664#
L1FC0     3671#     3703      3703
```

```
L1FDF     3683#    3707     3707
L2005     3697#    3774     3777     3852     3855
L2023     3705#    3712
L2040     3699     3701     3713#
L20C3     3766#    3770
L210A     3799     3803#
L210D     3803     3805#
L2177     3843#    3847
L21A0     2456     3868#
LC40        36#    3188
LD9        163#    1045
LDE        167#    3124
LDF        168#    3133
LDORG       40#    1024
LE0        169#    2224     2240     2277     2283     2290
LE1        170#    2225     2239     2245     2249     2253     2258     2262     2268     2273     2299     2303
LFCHAR      69#    1539     1552     2309     2927
LINCNT     182#    3160     3307     3308     3319     3319     3439     3440
LOCVAR     136#    1389     1391     1427     1429     1719     1719     2090     2092     2104     2106     2119
          2124     3735     3735     3815     3815
LRUPAG     144#    1065     1124     2858     2872
LSTFLC      55#    3166     3166
MAINOR      51#      52     1025
MDFLAG     178#    2521     2527     2540     3807     3848
MNLOOP    1127     1222#    1287
MOREMS    3303#    3305     3311     3311
MOV        507#    1039     1041     1044     1053     1061     1063     1064     1073     1075     1083     1088
          1099     1101     1102     1105     1110     1114     1125     1222     1259     1260     1283     1285
          1310     1335     1351     1388     1390     1401     1403     1413     1426     1428     1439     1441
          1464     1484     1506     1514     1521     1523     1527     1529     1558     1560     1562     1564
          1570     1583     1609     1618     1619     1684     1687     1698     1700     1712     1719     1725
          1727     1738     1739     1741     1798     1801     1869     1874     1892     1916     1918     1924
          1926     1929     1930     2063     2064     2073     2089     2091     2103     2105     2111     2114
          2115     2135     2158     2160     2179     2204     2207     2210     2220     2222     2223     2224
          2236     2280     2284     2286     2294     2337     2339     2355     2357     2402     2426     2520
          2563     2565     2574     2576     2616     2654     2656     2663     2665     2711     2713     2719
          2729     2730     2739     2740     2741     2742     2809     2811     2817     2827     2828     2842
          2845     2846     2848     2851     2855     2857     2883     2884     2956     2958     2967     2970
          3007     3024     3027     3039     3042     3043     3046     3144     3147     3150     3151     3152
          3153     3154     3159     3312     3314     3318     3353     3355     3366     3422     3438     3441
          3452     3495     3598     3602     3613     3628     3646     3647     3655     3658     3713     3714
          3724     3727     3763     3782     3806     3812     3840     3860
MRKPAG    2702     2800     2838#
MRMSLN    3305#    3311
MRUPAG     143#    1064     2839     2839     2840     2852
NEWMOD    2891     2930#
OPADD     1197     1974#
OPAND     1186     1811#
OPCALL    1207     2058#
```

```
OPCGPA    1228     1324#
OPCGPB    1229     1304#
OPCGPC    1230     1292#
OPCGPD    1235#
OPCKSM    1148     1557#
OPCLRA    1189     1845#
OPCODE     115#    1225     1275     1313     1329     1337
OPCRLF    1146     1550#
OPDEC     1160     1643#    1780
OPDECB    1181     1780#
OPDIV     1200     2005#
OPDSTT    1163     1659#    1864
OPENDS    1145     3899#
OPGTBY    1193     1888#
OPGTCH    1156     1604#
OPGTLN    1211     2215#
OPGTNP    1196     1957#
OPGTP     1194     1899#
OPGTPA    1195     1937#
OPGTPL    1158     1622#
OPGTPR    1157     1615#
OPGTSB    1155     1599#
OPGTWD    1192     1882#
OPINC     1159     1631#    1784
OPINCB    1182     1784#
OPJUMP    1166     1750#
OPMAX1    1149#    1294
OPMAX2    1170#    1316
OPMAX3    1202#    1340
OPMAX4    1217#    1280
OPMOVE    1168     1760#
OPMOVT    1191     1864#
OPMTCH    1178     2033#
OPMUL     1199     1986#
OPNOT     1169     1765#
OPNULL    1139     1532#
OPOR      1185     1808#
OPPRCH    1212     2412#
OPPRNM    1213     2418#
OPPRST    1147     2215     3351#
OPPRTN    1164     1694#
OPPSB     1161     1651#
OPPSI     1137     1521#    1535
OPPSIC    1138     1535#
OPPSW     1167     1755#
OPPTBY    1209     2168#
OPPTP     1210     2187#
OPPTWD    1208     2145#
OPPULL    1216     2472#
```

```
OPPUSH   1215    2466#
OPRMD    1201    2022#
OPRNDM   1214    2455#
OPRSGM   1141    3790#
OPRTN    1165    1515    1547    1711#
OPRTNF   1136    1480    1517#
OPRTNT   1135    1482    1512#   1542
OPRTNV   1143    1545#
OPSETA   1188    1831#
OPSUB    1198    1980#
OPSVGM   1140    3720#
OPTAB1   1135#   1149    1296    1296
OPTAB2   1154#   1170    1318    1318
OPTAB3   1177#   1202    1342    1342
OPTAB4   1207#   1217    1275    1275
OPTINT   1183    1791#
OPTSTA   1187    1818#
OPTSTZ   1154    1596#
OUTBUF   3236#   3321    3351    3437    3625    3645    3709
OUTMSG   3527#   3596    3624    3644    3704    3708    3894    3903
PKWORD    165#   2368    2384    2388    2392    2396    3040    3055    3063    3075    3084    3113
         3117    3119    3120    3121    3122    3123    3128    3130    3131    3132    3134    3135
         3137
PNYBBF    159#   2961    2961    2969    2969    3000    3000    3000    3008    3010    3013    3025
         3028
PNYBCN    158#   2884    2960    2970    2992    2997    3008    3025    3028
PRCSWL    183#   3145    3252    3252    3279    3279    3332    3332    3335    3335
PREDFL   1450#   1592    1597    1613    1772    1778    1789    1796    1806    1826    2050    3778
         3856
PREDTR   1446#   1591    1597    1789    2051    3775    3853
PRGIDX    123#   1100    1491    1493    1522    1528    1739    2065    2076    2076    2678    2682
         3731    3731    3810    3810
PRGLPG    124#   1102    1103    1499    1500    1502    1504    1523    1523    1529    1529    1733
         1733    2071    2071    2078    2688    2688    2690    2692    2696    2696    2725    2725
         2730    2731
PRGMPT    125#   1531    1531    2679    2711    2712
PRGPPG    127#   2697    2705    2724    2728    2817
PRGUPD    126#   1040    1507    1530    1742    2074    2675    2686    2714    2818    3813
PRMMOD    157#   2884    2936    2938    2939    2960    2970    2984
PRNTBF   3185    3216    3307#   3592    3593    3597    3621    3635    3640    3701    3705    3712
         3890    3899    3905
PRNTNM   2419    2424#   3375    3393    3404    3419    3897
PRNTST   1525    1706    1757    2883#   2964
PROMPT     86#   3154
PRTBUF   3240    3248#   3451
PRTWDT     48#   3262
PSH       485#   1376    1376    1417    1417    1607    1635    1635    1666    1666    1673    1673
         1684    1684    1868    1868    2276    2430    2478    2478    2498    2498    2959    2961
         2961    2961    2963    2963    3249    3249    3249    3320    3331    3331    3352
```

```
PTVARA    1370    1421#
PTVARP    1414    1416#   1610
PTVRA1    1369#   1636    1862
PTVRP1    1414#   1620    1744    1762    1766    1809    1812    1886    1893    1919    1928    1931
          1951    1975    1981    1996    2027    2060    2460
PTVRPA    1412#   1627    1966
PTVRPZ    1411#   1941    1963
PUL        478#   1361    1378    1378    1420    1420    1472    1638    1638    1672    1672    1683
          1683    1686    1686    1876    1876    2100    2131    2293    2491    2491    2518    2518
          2766    2966    2966    2966    2969    2969    2969    3280    3282    3282    3336    3336
          3423    3897    3897
PULLWD    1144    1372    1376    1406    1545    1714    1723    1731    1734    1737    2472    2662#
PUSHWD    1373    1378    1422    2065    2068    2071    2095    2132    2467    2652#
RDKEY       98#   3315    3626    3656    3710
RNDLOC      90#   3868    3869    3871    3871
RNGDBG      35#    103    2453    3146
RSGMFL    3794    3803    3823    3831    3844    3855#
RSGMMV    3811    3816    3820    3828    3836    3859#   3863
RTSCC      699#
RTSCS      706#   2302    2313    2886
RTSEQ      685#   1663    2229    2230    2298    2529
RTSGE      720#
RTSGT      727#   3223
RTSLT      713#
RTSMI      742#
RTSNE      692#   2683    2781
RTSPL      735#   2540
RWTS        63#   3508
RWTSOR      53#     54      63
SBWDPT     172#   1111    1113    2957    2959
SCMSLN    3343#   3370
SCORMS    3341#   3343    3370    3370
SECPTK     113#   3499    3506
SEPTAB    2308#   2317
SETAXB    1571    1652    1705    1884    1889    2739#
SETAXW    1756    2748#   2963
SETUPA    1818    1831    1845    2547#
SETUPP    1899    1937    1957    2187    2571#
SETUPT    1600    1605    1616    1660    1667    1678    1695    1792    1866    1870    2548    2572
          2615#
SHWMSG    3313    3371    3381    3427#   3650    3654
SPCLCH    2906    2973#   3093
START     1027#   1142    3911
STCKLC      45#     46    1043    1043
STCKMX      44#     46    2660
STKCNT     151#   1042    1713    2136    2657    2658    2667    3739    3739    3819    3819
STKCSV     154#   1713    1740    2064    2136
STKLIM      46#   3747    3747    3755    3755    3827    3827    3835    3835
STKPNT     152#   1043    1043    1712    1712    2137    2137    2653    2653    2653    2655    2656
```

```
          2656    2656    2657    2664    2665    2665    2666    2667    2667
STKPSV     153#   1712    1712    1736    1736    2068    2068    2137    2137
STLTYP     187#   1096    3367    3398
STR        526#
SUB        370#   1057    1118    1292    1385    1393    1424    1431    1481    1483    1624    1646
          1729    1752    2108    2401    2551    2556    2653    2656    2901    2930    2947    3007
          3061    3061    3068    3193    3316
SVGMFL    3743    3751    3759    3767    3772    3777#
SVGMMV    3732    3736    3740    3748    3756    3781#   3785
SWPMEM     138#   1115    1116    1119    2723    2723    2821    2821
SWPPGS     141#   1122    2863
TBCHAR      70#   2309
THGATT     218#
THGCHD     221#   1606    1668    1675    1871
THGPAR     219#   1617    1661    1686    1868
THGPRP     222#   1697    2573
THGSIB     220#   1601    1673    1679    1878
TIMEMS    3345#   3347    3380    3380
TMMSLN    3347#   3380
TMPMOD     156#   2885    2934    2971    2982    2987
TSTA       522#   1366    1370    1384    1422    1448    1452    1613    2084    2540    2893    3053
          3109
TSTABE     909#   1365    1369    1383    2083    3052    3108
TSTABM     924#   1447
TSTABN     914#   1612    2892
TSTABP     919#   1451
TSTAJE     929#   1421
TSTARP     934#   2539
TSTCHR    3051    3059    3100#
TSTMOD    2891    2932    2982#
VERSN       34#     50
VMT1LC      58#   1047    1047
VMT2LC      59#   1048    1048
VMT3LC      60#   1049    1049
VMT4LC      61#   1050    1050
VMTAB1     146#   1047    1047    1054    2730    2828    2867
VMTAB2     147#   1048    1048    1054    2731    2829    2869
VMTAB3     148#   1049    1049    1056    1062    1126    2843    2844    2849
VMTAB4     149#   1050    1050    1058    2846    2847    2852    2856
VMTORG      52#     53      58      59      60      61
VTAB        95#   3354    3424
WNDBOT      79#   3153    3310
WNDLFT      76#   3151
WNDTOP      78#   3148    3160    3319    3439
WNDWDT      77#   3152    3199    3211    3223    3324
ZPORG       42#    111
```