```
                   ; PROCEDURE SCREEN UPDATE
                   ; THIS PROCEDURE UPDATES THE POSITION OF ALL OBJECTS
                   ; IN THE OBJECT OBLIST
                   ;
                   ;
                   ;
                   ; GLOBALS - FIGPTR,PIXLEN,BYTWID,XPOS,YPOS,OBLIST,APTR,DPTR,STRLST
                   ;
                   ;
                   ;
AFD4 A9 FF         UPDATE  LDA     #$FF
AFD6 20 61 AD              JSR     SCHED
                   ;
AFD9 A9 00         UPDINI  LDA     #0
AFDB 8D 04 06              STA     TASK1+4 MAKE TASK 1 AND 2 ACTIVE
AFDE 8D 24 06              STA     TASK2+4
AFE1 A5 86                 LDA     STRLST  SETUP RESTORE OBLIST PTR
AFE3 85 90                 STA     APTR
AFE5 A5 87                 LDA     STRLST+1
AFE7 85 91                 STA     APTR+1
                   ; ERASES ALL OBJECTS AT OLD POSITIONS
AFE9 A0 00         ERASLP  LDY     #0      WIDTH
AFEB B1 90                 LDA     (APTR),Y
AFED F0 29                 BEQ     DROP    BRANCH IF TERMINATOR
AFEF 8D 94 07              STA     BYTWID
AFF2 A0 01                 LDY     #1      LENGTH
AFF4 B1 90                 LDA     (APTR),Y
AFF6 8D 90 07              STA     PIXLEN
AFF9 A0 02                 LDY     #2      XPOS
AFFB B1 90                 LDA     (APTR),Y
AFFD 8D 95 07              STA     XPOS
B000 A0 03                 LDY     #3      YPOS
B002 B1 90                 LDA     (APTR),Y
B004 8D 96 07              STA     YPOS
B007 20 CC AC              JSR     SRESTO
                   ;
B00A A5 90                 LDA     APTR
B00C 18                    CLC
B00D 69 04                 ADC     #4
B00F 85 90                 STA     APTR
B011 90 02                 BCC     *+4
B013 E6 91                 INC     APTR+1
B015 4C E9 AF              JMP     ERASLP
                   ;
                   ; PLACE ALL OBJECTS AT NEW POSITIONS
B018 8D BA 07      DROP    STA     OBNO    (0)
                   ;
B01B AD EB 09              LDA     OBLIST
B01E 85 92                 STA     DPTR
B020 AD EC 09              LDA     OBLIST+1
B023 85 93                 STA     DPTR+1
B025 A5 86                 LDA     STRLST
B027 85 90                 STA     APTR
B029 A5 87                 LDA     STRLST+1
B02B 85 91                 STA     APTR+1
                   ;
                   ; GET NEW X POSITION
B02D A0 08         DROPLP  LDY     #NEWX
B02F B1 92                 LDA     (DPTR),Y
B031 8D 95 07              STA     XPOS
```

```
                              ; UPDATE OBJECT OLDX
B034 A0 06                          LDY     #OLDX
B036 91 92                          STA     (DPTR),Y
B038 A0 02                          LDY     #2          XPOS
B03A 91 90                          STA     (APTR),Y
                              ;
                              ; CALC OFFSET HERE!!!!!
                              ;
                              ; GET NEW Y POSITION
B03C A0 09                          LDY     #NEWY
B03E B1 92                          LDA     (DPTR),Y
B040 8D 96 07                       STA     YPOS
                              ; UPDATE OBJECT OLDY
B043 A0 07                          LDY     #OLDY
B045 91 92                          STA     (DPTR),Y
B047 A0 03                          LDY     #3          YPOS
B049 91 90                          STA     (APTR),Y
                              ;
                              ;
                              ;
                              ; COLOR TRANSFORM GOES HERE!!!!!
                              ;
                              ; SET FIGURE PTR TO START OF PIC BLOCK
B04B A0 04                          LDY     #PICADD
B04D B1 92                          LDA     (DPTR),Y
B04F 85 80                          STA     FIGPTR
B051 C8                             INY
B052 B1 92                          LDA     (DPTR),Y
B054 85 81                          STA     FIGPTR+1
                              ; SEQUENCER GOES HERE
B056 AE BA 07                       LDX     OBNO
B059 BD BB 07                       LDA     OBJFLG,X
B05C C9 01                          CMP     #1
B05E F0 34                          BEQ     SEQ1        FIRST TIME THRU
B060 BD A9 07                       LDA     ANMRAT,X
B063 F0 15                          BEQ     ABYPAS
B065 DE A9 07                       DEC     ANMRAT,X
B068 D0 10                          BNE     ABYPAS
B06A A9 04                          LDA     ~~#4~~ ARATE,X
B06C 9D A9 07                       STA     ANMRAT,X
B06F BD B1 07                       LDA     ANM,X
B072 18                             CLC
B073 69 10                          ADC     #16         ANM OFFSET          CMP   ALIMIT,X
B075 29 3F                          AND     #$3F                            BCC   *+4
B077 9D B1 07                       STA     ANM,X                           LDA   #0
B07A BD AD 07         ABYPAS        LDA     ORNRAT,X
B07D F0 15                          BEQ     SEQ1
B07F DE AD 07                       DEC     ORNRAT,X
B082 D0 10                          BNE     SEQ1
B084 A9 00                          LDA     #0
B086 9D AD 07                       STA     ORNRAT,X
B089 BD B5 07                       LDA     ORN,X
B08C 18                             CLC
B08D 69 02                          ADC     #2          ORN OFFSET
B08F 29 0F                          AND     #$0F
B091 9D B5 07                       STA     ORN,X
B094 A9 00            SEQ1          LDA     #0
B096 9D BB 07                       STA     OBJFLG,X
B099 18                             CLC
B09A 7D B1 07                       ADC     ANM,X
```

```
B09D 18                    CLC
B09E 7D B5 07              ADC     ORN,X
B0A1 A8                    TAY
                   ;
B0A2 B1 80                 LDA     (FIGPTR),Y
B0A4 AA                    TAX
B0A5 C8                    INY
B0A6 B1 80                 LDA     (FIGPTR),Y
B0A8 F0 39                 BEQ     NEXTOB
B0AA 85 81                 STA     FIGPTR+1
B0AC 86 80                 STX     FIGPTR
                   ; GET OFFSET BYTE WIDTH AND SAVE IN OBJ BLOCK
B0AE A0 02                 LDY     #2
B0B0 B1 80                 LDA     (FIGPTR),Y
B0B2 8D 8F 07              STA     PIXWID
B0B5 AD 95 07              LDA     XPOS
B0B8 29 03                 AND     #3
B0BA 18                    CLC
B0BB 6D 8F 07              ADC     PIXWID
B0BE 18                    CLC
B0BF 69 03                 ADC     #3
B0C1 4A                    LSR     A
B0C2 4A                    LSR     A
B0C3 8D 94 07              STA     BYTWID
B0C6 A0 00                 LDY     #0      WIDTH
B0C8 91 90                 STA     (APTR),Y
                   ; GET OFFSET PIXEL LENGTH
                   ; AND SAVE IN OBJ BLOCK
B0CA A0 03                 LDY     #3      GET LENGTH
B0CC B1 80                 LDA     (FIGPTR),Y
B0CE 8D 90 07              STA     PIXLEN
B0D1 A0 01                 LDY     #1      LENGTH
B0D3 91 90                 STA     (APTR),Y
B0D5 A5 80                 LDA     FIGPTR
B0D7 18                    CLC
B0D8 69 04                 ADC     #4
B0DA 85 80                 STA     FIGPTR
B0DC 90 02                 BCC     *+4
B0DE E6 81                 INC     FIGPTR+1
B0E0 20 54 AE              JSR     PLACE
                   ;
B0E3 A5 90         NEXTOB  LDA     APTR
B0E5 18                    CLC
B0E6 69 04                 ADC     #4
B0E8 85 90                 STA     APTR
B0EA 90 02                 BCC     *+4
B0EC E6 91                 INC     APTR+1
B0EE A0 00                 LDY     #NXTOBJ
B0F0 B1 92                 LDA     (DPTR),Y
B0F2 AA                    TAX
B0F3 C8                    INY
B0F4 B1 92                 LDA     (DPTR),Y
B0F6 85 93                 STA     DPTR+1
B0F8 86 92                 STX     DPTR
B0FA F0 06                 BEQ     *+8
B0FC EE BA 07              INC     OBNO
B0FF 4C 2D B0              JMP     DROPLP
B102 A0 00                 LDY     #0      ADD ROBLIST TERMINATOR
B104 98                    TYA
B105 91 90                 STA     (APTR),Y
```

```
B107 A9 01              LDA     #1
B109 8D A6 07           STA     UPDFLG
B10C 4C D4 AF           JMP     UPDATE
```

```
                               ; PROCEDURE STICK READ

                               ; THIS PROCEDURE READS THE PROPER JOYSTICK AND
                               ; UPDATES THE CURRENT TASK'S XPOS AND YPOS. IT ALSO
                               ; UPDATES THE LOCAL OBJECT'S NEWX AND NEWY.
B10F                                   P=*
                                       *=V
09F8                           JOY12   *=*+1
09F9                           EX      *=*+1
09FA                           WHY     *=*+1
09FB                           HDELTA  *=*+1
09FC                           VDELTA  *=*+1
09FD                                   V=*
                                       *=P
                               ;
0017                           TCOUNT  = 23             LOCAL VARIABLE IN TCB.
                               ;
B10F A0 10                     STKINI  LDY     #TXPOS
B111 B1 88                             LDA     (CURTSK),Y
B113 A2 00                             LDX     #0
B115 A0 06                             LDY     #OLDX
B117 20 B7 AD                          JSR     QUEATT
B11A A0 11                             LDY     #TYPOS
B11C B1 88                             LDA     (CURTSK),Y
B11E A2 00                             LDX     #0
B120 A0 07                             LDY     #OLDY
B122 20 B7 AD                          JSR     QUEATT

B125 A0 16                     STKLP   LDY     #DELTAT
B127 B1 88                             LDA     (CURTSK),Y
B129 A0 17                             LDY     #TCOUNT SET UP TCOUNT
B12B 91 88                             STA     (CURTSK),Y
B12D A0 10                             LDY     #TXPOS  GET TCB X POSITION
B12F B1 88                             LDA     (CURTSK),Y
B131 8D F9 09                          STA     EX
B134 A0 11                             LDY     #TYPOS
B136 B1 88                             LDA     (CURTSK),Y
B138 8D FA 09                          STA     WHY
B13B A0 14                             LDY     #DELTAX
B13D B1 88                             LDA     (CURTSK),Y
B13F 8D FB 09                          STA     HDELTA
B142 A0 15                             LDY     #DELTAY
B144 B1 88                             LDA     (CURTSK),Y
B146 8D FC 09                          STA     VDELTA
                               ;
B149 A0 12                     IENTER  LDY     #STKNO
B14B B1 88                             LDA     (CURTSK),Y
B14D F0 0A                             BEQ     STK0
B14F AD 00 D3                  STK1    LDA     PORTA
B152 4A                                LSR     A
B153 4A                                LSR     A
B154 4A                                LSR     A
B155 4A                                LSR     A
B156 4C 5E B1                          JMP     STOJOY
B159 AD 00 D3                  STK0    LDA     PORTA
B15C 29 0F                             AND     #$0F
B15E 8D F8 09                  STOJOY  STA     JOY12
B161 C9 0F                             CMP     #$0F
B163 D0 03                             BNE     *+5
B165 4C C3 B1                          JMP     JDON
B168 A0 00                             LDY     #0       KEEP ATRACT OFF
```

```
   0 1 2 3
0  BEQ
   CMP #2
2  BEQ
1  BCC
   1

        4

   0, 1, 2, 3
```

```
      CMP  #2
      BEQ  STK2
      BCC  STK1
STK3  LDA  PORTB
      AND  #$0F
      JMP  STOJOY
STK2  LDA  PORTB
      LSR  A
      LSR  A
      LSR  A
      LSR  A
      JMP  STOJOY
```

~~JMP STK0~~
~~JMP STK1~~
~~JMP STK2~~
~~JMP STK3~~

```
B16A 84 4D                 STY     ATRACT
B16C 29 04                 AND     #4
B16E D0 11                 BNE     CHKRT
B170 AD F9 09              LDA     EX
B173 38                    SEC
B174 ED FB 09              SBC     HDELTA
B177 C9 A0                 CMP     #160
B179 B0 03                 BCS     *+5
B17B 8D F9 09              STA     EX
B17E 4C 96 B1              JMP     CHKUP
B181 AD F8 09      CHKRT   LDA     JOY12
B184 29 08                 AND     #8
B186 D0 0E                 BNE     CHKUP
B188 AD F9 09              LDA     EX
B18B 18                    CLC
B18C 6D FB 09              ADC     HDELTA
B18F C9 A0                 CMP     #160
B191 B0 03                 BCS     *+5
B193 8D F9 09              STA     EX
B196 AD F8 09      CHKUP   LDA     JOY12
B199 29 01                 AND     #1
B19B D0 11                 BNE     CHKDWN
B19D AD FA 09              LDA     WHY
B1A0 38                    SEC
B1A1 ED FC 09              SBC     VDELTA
B1A4 C9 50                 CMP     #SCNEND
B1A6 B0 03                 BCS     *+5
B1A8 8D FA 09              STA     WHY
B1AB 4C C3 B1              JMP     JDON
B1AE AD F8 09      CHKDWN  LDA     JOY12
B1B1 29 02                 AND     #2
B1B3 D0 0E                 BNE     JDON
B1B5 AD FA 09              LDA     WHY
B1B8 18                    CLC
B1B9 6D FC 09              ADC     VDELTA
B1BC C9 50                 CMP     #SCNEND
B1BE B0 03                 BCS     *+5
B1C0 8D FA 09              STA     WHY
                   ;
B1C3 AD F9 09      JDON    LDA     EX
B1C6 A0 10                 LDY     #TXPOS
B1C8 91 88                 STA     (CURTSK),Y
B1CA A2 00                 LDX     #0
B1CC A0 08                 LDY     #NEWX
B1CE 20 B7 AD              JSR     QUEATT
B1D1 AD FA 09              LDA     WHY
B1D4 A0 11                 LDY     #TYPOS
B1D6 91 88                 STA     (CURTSK),Y
B1D8 A2 00                 LDX     #0
B1DA A0 09                 LDY     #NEWY
B1DC 20 B7 AD              JSR     QUEATT
                   ;
B1DF A9 FF         STWAIT  LDA     #$FF
B1E1 20 61 AD              JSR     SCHED   RETURN TO SCHEDULER

B1E4 A0 17                 LDY     #TCOUNT
B1E6 B1 88                 LDA     (CURTSK),Y
B1E8 38                    SEC
B1E9 E9 01                 SBC     #1
B1EB 91 88                 STA     (CURTSK),Y
```

```
B1ED D0 F0                    BNE     STWAIT
B1EF 4C 25 B1                 JMP     STKLP
```

```
                   ;
                   ; MEMORY MANAGEMENT PACKAGE
                   ;
                   ; AVAILABLE MEMORY IS DIVIDED INTO TWO REGIONS WHICH GROW TOWARD EACH
                   ; OTHER; THE REGIONS ARE DEFINED BY FOUR POINTER VARIABLES:
                   ;
                   ;       'S1L' POINTS TO BOTTOM OF REGION #1
                   ;       'S1H' POINTS TO FIRST UNUSED LOCATION ABOVE REGION #1
                   ;       'S2L' POINTS TO BOTTOM OF REGION #2
                   ;       'S2H' POINTS TO FIRST UNUSED LOCATION ABOVE REGION #2
                   ;
                   ; THREE ROUTINES ARE PROVIDED TO ALLOCATE AND DEALLOCATE MEMORY:
                   ;
                   ;       'MINIT' IS USED TO INITIALIZE MEMORY
                   ;       'MALLOC' IS USED TO ALLOCATE MEMORY
                   ;       'MDEALL' IS USED TO DEALLOCATE MEMORY
                   ;
                   ; THE TWO REGIONS ARE MAINTAINED AS TWO COMPRESSED STACKS; ALLOCATION
                   ; AND DEALLOCATION INVOLVES THE MOVEMENT OF DATA TO CREATE AND
                   ; ELIMINATE HOLES IN THE STACKS.
                   ;


B1F2 A2 02         MINIT   LDX     #S1H-DTAB       ; S1H <= S1L
B1F4 A0 00                 LDY     #S1L-DTAB
B1F6 20 4A B3              JSR     DMOVI

B1F9 A2 04                 LDX     #S2L-DTAB       ; S2L <= S2H.
B1FB A0 06                 LDY     #S2H-DTAB
B1FD 20 4A B3              JSR     DMOVI

B200 60                    RTS



                   ;
                   ; MALLOC -- MEMORY ALLOCATE
                   ;
                   ; CALLING SEQUENCE:
                   ;
                   ;       'MEMA' CONTAINS THE ADDRESS OF THE START OF ALLOCATION
                   ;          REGION #1: DATA AT START ADDRESS AND ABOVE ARE MOVED UP.
                   ;          REGION #2: DATA BELOW START ADDRESS ARE MOVED DOWN.
                   ;       'MEMB' CONTAINS THE NUMBER OF BYTES TO ALLOCATE
                   ;
                   ;       JSR     MALLOC
                   ;       BNE     NOT ENOUGH MEMORY TO SATISFY ALLOCATION
                   ;
                   ;       'MEMA' CONTAINS LOWEST ADDRESS IN THE ALLOCATED BLOCK
                   ;       FIRST TWO BYTES OF ALLOCATED BLOCK = BLOCK SIZE
                   ;
B201 A0 02         MALLOC  LDY     #S1H-DTAB       ; ACC = S1H ...
B203 20 40 B4              JSR     DLOADA

B206 A0 0A                 LDY     #MEMB-DTAB      ; ... + MEMB.
B208 20 4A B4              JSR     DADDA

B20B A0 04                 LDY     #S2L-DTAB       ; COMPARE ACC WITH S2L.
B20D 20 54 B4              JSR     DCMPA
```

```
B210 B0 6A                  BCS   MAL300           ; NOT ENOUGH ROOM.

B212 A2 08                  LDX   #MEMA-DTAB       ; SEE IF ALLOCATION IN REGION #1 OR #2.
B214 A0 04                  LDY   #S2L-DTAB
B216 20 3D B3               JSR   DCMPI
B219 B0 28                  BCS   MAL100           ; REGION #2.

                    ; ALLOCATE FROM REGION #1

B21B A2 0C                  LDX   #MSP-DTAB        ; MSP = MEMA.
B21D A0 08                  LDY   #MEMA-DTAB
B21F 20 4A B3               JSR   DMOVI

B222 A2 0E                  LDX   #MDP-DTAB        ; MDP = MEMA ...
B224 20 4A B3               JSR   DMOVI

B227 A0 0A                  LDY   #MEMB-DTAB       ; ... + MEMB.
B229 20 55 B3               JSR   DADDI

B22C A2 10                  LDX   #MBC-DTAB        ; MBC = S1H ...
B22E A0 02                  LDY   #S1H-DTAB
B230 20 4A B3               JSR   DMOVI

B233 A0 08                  LDY   #MEMA-DTAB       ; ... - MEMA.
B235 20 65 B3               JSR   DSUBI

B238 A2 02                  LDX   #S1H-DTAB        ; S1H = ACC (= S1H + MEMB).
B23A 20 45 B4               JSR   DSTORA

B23D 20 10 B3               JSR   MOVDA            ; MOVE DATA UPWARD.

B240 4C 6E B2               JMP   MAL200

                    ; ALLOCATE IN REGION #2

B243 A2 0C          MAL100  LDX   #MSP-DTAB        ; MSP = S2L.
B245 A0 04                  LDY   #S2L-DTAB
B247 20 4A B3               JSR   DMOVI

B24A A2 10                  LDX   #MBC-DTAB        ; MBC = MEMA ...
B24C A0 08                  LDY   #MEMA-DTAB
B24E 20 4A B3               JSR   DMOVI

B251 A0 04                  LDY   #S2L-DTAB        ; ... - S2L.
B253 20 65 B3               JSR   DSUBI

B256 A2 04                  LDX   #S2L-DTAB        ; S2L = S2L - MEMB.
B258 A0 0A                  LDY   #MEMB-DTAB
B25A 20 65 B3               JSR   DSUBI

B25D A2 0E                  LDX   #MDP-DTAB        ; MDP = S2L (NEW VALUE).
B25F A0 04                  LDY   #S2L-DTAB
B261 20 4A B3               JSR   DMOVI

B264 A2 08                  LDX   #MEMA-DTAB       ; MEMA = MEMA - MEMB.
B266 A0 0A                  LDY   #MEMB-DTAB
B268 20 65 B3               JSR   DSUBI

B26B 20 EB B2               JSR   MOVIA            ; MOVE DATA DOWNWARD.
```

```
B26E A0 00          MAL200  LDY     #0              ; MOVE BLOCK SIZE TO BLOCK.
B270 A5 A0                  LDA     MEMB
B272 91 9E                  STA     (MEMA),Y
B274 C8                     INY
B275 A5 A1                  LDA     MEMB+1
B277 91 9E                  STA     (MEMA),Y

B279 A9 00                  LDA     #0              ; SET CC FOR NORMAL EXIT.
B27B 60                     RTS

B27C A9 FF          MAL300  LDA     #$FF            ; SET CC FOR ERROR EXIT.
B27E 60                     RTS



                    ;
                    ; MDEALL -- MEMORY DEALLOCATE
                    ;
                    ; CALLING SEQUENCE:
                    ;
                    ;           'MEMA' = ADDRESS OF BLOCK TO DEALLOCATE
                    ;           FIRST 2 BYTES OF BLOCK = SIZE OF BLOCK
                    ;
                    ;           JSR     MDEALL
                    ;
                    ;           'MEMA' = ADDRESS OF BLOCK FOLLOWING DEALLOCATED BLOCK (AFTER DEALL)
                    ;
B27F A0 00          MDEALL  LDY     #0              ; GET SIZE OF BLOCK TO MEMB.
B281 B1 9E                  LDA     (MEMA),Y
B283 85 A0                  STA     MEMB
B285 C8                     INY
B286 B1 9E                  LDA     (MEMA),Y
B288 85 A1                  STA     MEMB+1

B28A A2 08                  LDX     #MEMA-DTAB      ; SEE IF IN REGION #1 OR #2.
B28C A0 04                  LDY     #S2L-DTAB
B28E 20 3D B3               JSR     DCMPI
B291 B0 2C                  BCS     MDA100          ; REGION #2.

                    ; DEALLOCATE FROM REGION #1.

B293 A2 0C                  LDX     #MSP-DTAB       ; MSP = MEMA ...
B295 A0 08                  LDY     #MEMA-DTAB
B297 20 4A B3               JSR     DMOVI

B29A A0 0A                  LDY     #MEMB-DTAB      ; ... + MEMB.
B29C 20 55 B3               JSR     DADDI

B29F A2 10                  LDX     #MBC-DTAB       ; MBC = S1H ...
B2A1 A0 02                  LDY     #S1H-DTAB
B2A3 20 4A B3               JSR     DMOVI

B2A6 A0 0C                  LDY     #MSP-DTAB       ; ... - MSP.
B2A8 20 65 B3               JSR     DSUBI

B2AB A2 02                  LDX     #S1H-DTAB       ; S1H = S1H - MEMB.
B2AD A0 0A                  LDY     #MEMB-DTAB
B2AF 20 65 B3               JSR     DSUBI

B2B2 A2 0E                  LDX     #MDP-DTAB       ; MDP = MEMA.
```

```
B2B4 A0 08                  LDY    #MEMA-DTAB
B2B6 20 4A B3               JSR    DMOVI

B2B9 20 EB B2               JSR    MOVIA            ; MOVE DATA DOWNWARD.

B2BC 4C EA B2               JMP    MDA200

                  ; DEALLOCATE MEMORY IN REGION #2

B2BF A2 0C        MDA100    LDX    #MSP-DTAB        ; MSP = S2L.
B2C1 A0 04                  LDY    #S2L-DTAB
B2C3 20 4A B3               JSR    DMOVI

B2C6 A2 10                  LDX    #MBC-DTAB        ; MBC = MEMA ...
B2C8 A0 08                  LDY    #MEMA-DTAB
B2CA 20 4A B3               JSR    DMOVI

B2CD A0 04                  LDY    #S2L-DTAB        ; ... - S2L.
B2CF 20 65 B3               JSR    DSUBI

B2D2 A2 04                  LDX    #S2L-DTAB        ; S2L = S2L + MEMB.
B2D4 A0 0A                  LDY    #MEMB-DTAB
B2D6 20 55 B3               JSR    DADDI

B2D9 A2 0E                  LDX    #MDP-DTAB        ; MDP = S2L (NEW VALUE).
B2DB A0 04                  LDY    #S2L-DTAB
B2DD 20 4A B3               JSR    DMOVI

B2E0 A2 08                  LDX    #MEMA-DTAB       ; MEMA = MEMA + MEMB.
B2E2 A0 0A                  LDY    #MEMB-DTAB
B2E4 20 55 B3               JSR    DADDI

B2E7 20 10 B3               JSR    MOVDA            ; MOVE DATA UPWARD.

B2EA 60           MDA200    RTS
```

```
                        ;
                        ; MOVE UTILITIES FOR MEMORY MANAGEMENT
                        ;
                        ; MOVE BLOCKS OF DATA WITH EITHER INCREASING OR DECREASING ADDRESS
                        ;
                        ; THREE VARIABLES CONTROL THE MOVE ROUTINES:
                        ;
                        ;       'MSP' CONTAINS POINTER TO SOURCE DATA LOCATION
                        ;       'MDP' CONTAINS POINTER TO DESTINATION DATA LOCATION
                        ;       'MBC' CONTAINS THE NUMBER OF BYTES TO MOVE
                        ;
                        ;


                        ;
                        ; MOVIA -- MOVE DATA BLOCK WITH INCREASING ADDRESS
                        ;
                        ; CALLING SEQUENCE:
                        ;
                        ;       'MSP', 'MDP' & 'MBC' SETUP
                        ;
                        ;       JSR     MOVIA
                        ;
B2EB A5 A6              MOVIA   LDA     MBC             ; SEE IF BYTE COUNT = ZERO.
B2ED AA                         TAX                     ; SAVE LSB OF BYTE COUNT.
B2EE 05 A7                      ORA     MBC+1
B2F0 F0 1D                      BEQ     MVI090          ; ZERO -- NOTHING TO DO.

B2F2 A0 00                      LDY     #0              ; INDEX TO DATA BLOCK.

B2F4 B1 A2              MVI010  LDA     (MSP),Y         ; MOVE DATA.
B2F6 91 A4                      STA     (MDP),Y
B2F8 C8                         INY                     ; BUMP INDEX.
B2F9 D0 04                      BNE     MVI020          ; NO PAGE WRAP.

B2FB E6 A3                      INC     MSP+1           ; PAGE WRAP -- BUMP POINTER VARIABLES.
B2FD E6 A5                      INC     MDP+1

B2FF CA                 MVI020  DEX                     ; DONE?
B300 D0 04                      BNE     MVI030          ; NO.

B302 A5 A7                      LDA     MBC+1           ; NOT SURE -- CHECK FURTHER.
B304 F0 09                      BEQ     MVI090          ; YES -- DONE.

B306 E0 FF              MVI030  CPX     #$FF            ; MAINTAIN D.P. BYTE COUNT.
B308 D0 EA                      BNE     MVI010

B30A C6 A7                      DEC     MBC+1           ; BORROW FROM MSB.
B30C 4C F4 B2                   JMP     MVI010

B30F 60                 MVI090  RTS


                        ;
                        ; MOVDA -- MOVE DATA BLOCK WITH DECREASING ADDRESS
                        ;
                        ; CALLING SEQUENCE:
                        ;
                        ;       'MSP', 'MDP', & 'MBC' SETUP
                        ;       JSR     MOVDA
                        ;
    B310 A5 A6          MOVDA   LDA     MBC             ; SETUP BYTE COUNT ...
    B312 AA                     TAX
    B313 A8                     TAY                     ; ... AND DATA INDEX.
    B314 05 A7                  ORA     MBC+1           ; TEST FOR ZERO BYTE COUNT.
    B316 F0 24                  BEQ     MVD090          ; ZERO -- NOTHING TO DO.

    B318 18                     CLC                     ; ADJUST POINTERS FOR START.
    B319 A5 A3                  LDA     MSP+1
    B31B 65 A7                  ADC     MBC+1
    B31D 85 A3                  STA     MSP+1

    B31F 18                     CLC
    B320 A5 A5                  LDA     MDP+1
    B322 65 A7                  ADC     MBC+1
    B324 85 A5                  STA     MDP+1

    B326 88             MVD010  DEY                     ; DECREMENT INDEX.
    B327 C0 FF                  CPY     #$FF            ; WRAP?
    B329 D0 06                  BNE     MVD020          ; NO.

    B32B C6 A7                  DEC     MBC+1           ; YES -- DECREMENT ALL POINTERS (MSB).
    B32D C6 A3                  DEC     MSP+1
    B32F C6 A5                  DEC     MDP+1

    B331 B1 A2          MVD020  LDA     (MSP),Y         ; MOVE A DATA BYTE.
    B333 91 A4                  STA     (MDP),Y

    B335 CA                     DEX                     ; DONE?
    B336 D0 EE                  BNE     MVD010          ; NO -- CONTINUE.

    B338 A5 A7                  LDA     MBC+1           ; NOT SURE -- CHECK FURTHER.
    B33A D0 EA                  BNE     MVD010          ; NO -- CONTINUE.

    B33C 60             MVD090  RTS                     ; YES -- RETURN.
```

```
                   ;
                   ; DOUBLE PRECISION ROUTINES
                   ;
                   ; ALL VARIABLES ARE ACCESSED VIA THEIR OFFSET FROM SYMBOL 'DTAB'.
                   ; NORMALLY THE X AND/OR Y REGISTERS CONTAIN THE 'DTAB' OFFSET
                   ; VALUES TO THE VARIABLE(S) TO BE DEALT WITH.
                   ;



                   ;
                   ; DCMPI -- DOUBLE COMPARE INDEXED
                   ;
                   ; CALLING SEQUENCE:
                   ;
                   ;       X = DATA #1 OFFSET
                   ;       Y = DATA #2 OFFSET
                   ;
                   ;       JSR     DCMPI
                   ;
                   ;       CC = DTAB(X) : DTAB(Y)  (UNSIGNED)
                   ;
B33D B5 97         DCMPI   LDA     DTAB+1,X        ; COMPARE MSBS.
B33F D9 97 00              CMP     DTAB+1,Y
B342 D0 05                 BNE     DCM090          ; NOT EQUAL -- ALL DONE.

B344 B5 96                 LDA     DTAB,X          ; EQUAL -- COMPARE LSBS.
B346 D9 96 00              CMP     DTAB,Y

B349 60            DCM090  RTS



                   ;
                   ; DMOVI -- DOUBLE BYTE MOVE INDEXED
                   ;
                   ; CALLING SEQUENCE:
                   ;
                   ;       X = DESTINATION OFFSET
                   ;       Y = SOURCE OFFSET
                   ;
                   ;       JSR     DMOVI
                   ;
                   ;       DTAB(X) = DTAB(Y)
                   ;
B34A B9 96 00      DMOVI   LDA     DTAB,Y
B34D 95 96                 STA     DTAB,X

B34F B9 97 00              LDA     DTAB+1,Y
B352 95 97                 STA     DTAB+1,X

B354 60                    RTS



                   ;
                   ; DADDI -- DOUBLE PRECISION ADD
                   ;
                   ; CALLING SEQUENCE:
                          ;
                          ;      X = OFFSET TO
                          ;      Y = OFFSET TO
                          ;
                          ;      JSR     DADDI
                          ;
                          ;      DTAB(X) = DTAB(X) + DTAB(Y)
                          ;
B355 B5 96                DADDI  LDA     DTAB,X
B357 18                          CLC
B358 79 96 00                    ADC     DTAB,Y
B35B 95 96                       STA     DTAB,X

B35D B5 97                       LDA     DTAB+1,X
B35F 79 97 00                    ADC     DTAB+1,Y
B362 95 97                       STA     DTAB+1,X

B364 60                          RTS


                          ;
                          ; DSUBI -- DOUBLE PRECISION SUBTRACT
                          ;
                          ; CALLING SEQUENCE:
                          ;
                          ;      X = OFFSET
                          ;      Y = OFFSET
                          ;
                          ;      JSR     DSUBI
                          ;      BEQ     RESULT = 0
                          ;
                          ;      DTAB(X) = DTAB(X) - DTAB(Y)
                          ;
B365 B5 96                DSUBI  LDA     DTAB,X
B367 38                          SEC
B368 F9 96 00                    SBC     DTAB,Y
B36B 95 96                       STA     DTAB,X

B36D B5 97                       LDA     DTAB+1,X
B36F F9 97 00                    SBC     DTAB+1,Y
B372 95 97                       STA     DTAB+1,X

B374 15 96                       ORA     DTAB,X          ; SET CC FOR ZERO TEST.

B376 60                          RTS


                          ;
                          ; DMULI -- DOUBLE PRECISION MULTIPLY
                          ;
                          ; CALLING SEQUENCE:
                          ;
                          ;      X = OFFSET
                          ;      X = OFFSET
                          ;
                          ;      JSR     DMULI
                          ;
                          ;      DTAB(X) = DTAB(X) * DTAB(Y)
                          ;
```

```
B377 A9 10         DMULI   LDA     #16             ; SETUP LOOP COUNTER.
B379 8D 40 02              STA     TEMP+2

B37C A9 00                 LDA     #0              ; INITIALIZE TEMP ACCUMULATOR.
B37E 8D 3E 02              STA     TEMP
B381 8D 3F 02              STA     TEMP+1

B384 16 96         DMU010  ASL     DTAB,X          ; DOUBLE PRECISION SHIFT LEFT.
B386 36 97                 ROL     DTAB+1,X
B388 90 13                 BCC     DMU020          ; NO BIT PRESENT.

B38A 18                    CLC                     ; BIT SET -- ADD TO PARTIAL.
B38B AD 3E 02              LDA     TEMP
B38E 79 96 00              ADC     DTAB,Y
B391 8D 3E 02              STA     TEMP
B394 AD 3F 02              LDA     TEMP+1
B397 79 97 00              ADC     DTAB+1,Y
B39A 8D 3F 02              STA     TEMP+1

B39D CE 40 02      DMU020  DEC     TEMP+2          ; DONE?
B3A0 F0 09                 BEQ     DMU090          ; YES -- RESULT IS IN 'TEMP'.

B3A2 0E 3E 02              ASL     TEMP            ; NO -- DOUBLE PRECISION SHIFT LEFT.
B3A5 2E 3F 02              ROL     TEMP+1
B3A8 4C 84 B3              JMP     DMU010

B3AB AD 3E 02      DMU090  LDA     TEMP            ; DONE -- MOVE RESULT.
B3AE 95 96                 STA     DTAB,X
B3B0 AD 3F 02              LDA     TEMP+1
B3B3 95 97                 STA     DTAB+1,X
B3B5 60                    RTS


                   ;
                   ; DDIVI -- DOUBLE PRECISION DIVIDE
                   ;
                   ; CALLING SEQUENCE:
                   ;
                   ;       X = OFFSET TO DIVIDEND
                   ;       Y = OFFSET TO DIVISOR
                   ;
                   ;       JSR     DDIVI
                   ;
                   ;       DTAB(X) = DTAB(X) / DTAB(Y) (SIGNED)
                   ;       'TEMP' = REMAINDER (SIGN MAY BE WRONG!!!)
                   ;
B3B6 A9 11         DDIVI   LDA     #16+1           ; SETUP LOOP COUNTER.
B3B8 8D 40 02              STA     TEMP+2
B3BB 8E 41 02              STX     TEMP+3          ; SAVE INDEX TO DIVIDEND.

B3BE A9 00                 LDA     #0              ; INITIALIZE REMAINDER.
B3C0 8D 3E 02              STA     TEMP
B3C3 8D 3F 02              STA     TEMP+1

B3C6 B9 97 00              LDA     DTAB+1,Y        ; SEE IF DIVISOR IS NEGATIVE.
B3C9 8D 43 02              STA     TEMP+5
B3CC 10 0B                 BPL     DDI006          ; NO.

B3CE 20 1A B4              JSR     DNEGI           ; YES -- NEGATE DIVIDEND ...
```

```
B3D1 98                  TYA
B3D2 AA                  TAX
B3D3 20 1A B4            JSR     DNEGI          ; ... & DIVISOR.
B3D6 AE 41 02            LDX     TEMP+3         ; RESTORE INDEX TO DIVIDEND.

B3D9 BD 3F 02   DDI006   LDA     TEMP+1,X       ; SEE IF DIVIDEND IS NEGATIVE.
B3DC 8D 42 02            STA     TEMP+4
B3DF 10 03               BPL     DDI008         ; NO.

B3E1 20 1A B4            JSR     DNEGI          ; YES -- NEGATE IT NOW (& THEN AGAIN LATER).

B3E4 18         DDI008   CLC

B3E5 AE 41 02   DDI010   LDX     TEMP+3         ; GET INDEX TO DIVIDEND.
B3E8 36 96               ROL     DTAB,X         ; DOUBLE PRECISION ROTATE.
B3EA 36 97               ROL     DTAB+1,X

B3EC CE 40 02            DEC     TEMP+2         ; DONE?
B3EF F0 13               BEQ     DDI090         ; YES.

B3F1 2E 3E 02            ROL     TEMP           ; NO.
B3F4 2E 3F 02            ROL     TEMP+1

B3F7 A2 A8               LDX     #TEMP-DTAB     ; IS REMAINDER < DIVISOR?
B3F9 20 3D B3            JSR     DCMPI
B3FC 90 E7               BCC     DDI010         ; YES.

B3FE 20 65 B3            JSR     DSUBI          ; NO.
B401 38                  SEC
B402 B0 E1               BCS     DDI010         ; (BRA).

B404 AD 42 02   DDI090   LDA     TEMP+4         ; SEE IF RESULT IS NEGATIVE.
B407 10 03               BPL     DDI092         ; NO.

B409 20 1A B4            JSR     DNEGI          ; YES -- NEGATE POSITIVE RESULT.

B40C AD 43 02   DDI092   LDA     TEMP+5         ; WAS DIVISOR NEGATED EARLIER.
B40F 10 08               BPL     DDI095         ; NO.

B411 98                  TYA                    ; YES -- NEGATE IT BACK TO ORIGINAL SIGN.
B412 AA                  TAX
B413 20 1A B4            JSR     DNEGI
B416 AE 41 02            LDX     TEMP+3         ; RESTORE INDEX.

B419 60         DDI095   RTS


                ;
                ; DNEGI -- DOUBLE PRECISION NEGATE
                ;
                ; CALLING SEQUENCE:
                ;
                ;        X = OFFSET TO NUMBER
                ;
                ;        JSR     DNEGI
                ;
                ;        DTAB(X) = -DTAB(X)
                ;
B41A 38         DNEGI    SEC                    ; (CLEAR BORROW).
B41B A9 00               LDA     #0
```

```
B41D F5 96                 SBC     DTAB,X
B41F 95 96                 STA     DTAB,X

B421 A9 00                 LDA     #0
B423 F5 97                 SBC     DTAB+1,X
B425 95 97                 STA     DTAB+1,X
B427 60                    RTS


                   ;
                   ; DADDS -- ADD A REGISTER TO DOUBLE BYTE
                   ;
                   ; CALLING SEQUENCE:
                   ;
                   ;       A = SIGNED BINARY NUMBER (-128 TO 127)
                   ;       X = DTAB OFFSET TO DP NUMBER
                   ;
                   ;       JSR     DADDS
                   ;
                   ;       DTAB(X) = DTAB(X) + A
                   ;
B428 C9 00         DADDS   CMP     #0              ; SEE IF POSITIVE OR NEGATIVE.
B42A 30 0A                 BMI     DDA030          ; NEGATIVE.

B42C 18                    CLC                     ; POSITIVE -- ADD.
B42D 75 96                 ADC     DTAB,X
B42F 95 96                 STA     DTAB,X
B431 90 02                 BCC     DDA010          ; NO CARRY.

B433 F6 97                 INC     DTAB+1,X        ; CARRY -- ADD TO MSB.

B435 60            DDA010  RTS

B436 18            DDA030  CLC
B437 75 96                 ADC     DTAB,X
B439 95 96                 STA     DTAB,X
B43B B0 02                 BCS     DDA040          ; NO BORROW.

B43D D6 97                 DEC     DTAB+1,X        ; BORROW -- SUB FROM MSB.

B43F 60            DDA040  RTS
```

```
                   ;
                   ; ACCUMULATOR FUNCTIONS -- ASSUME THE EXISTENCE OF A DOUBLE PRECISION
                   ; VARIABLE WITHIN 'DTAB' NAMED 'ACC'.
                   ;

                   ;
                   ; DLOADA -- LOAD 'ACC' WITH DATA
                   ;
                   ; CALLING SEQUENCE:
                   ;
                   ;       Y = OFFSET TO SOURCE DATA
                   ;
                   ;       JSR     DLOADA
                   ;
                   ;       X = ACC OFFSET
                   ;       'ACC' = DTAB(Y)
                   ;
B440 A2 12         DLOADA  LDX     #ACC-DTAB
B442 4C 4A B3              JMP     DMOVI


                   ;
                   ; DSTORA -- STORE 'ACC' TO LOCATION
                   ;
                   ; CALLING SEQUENCE:
                   ;
                   ;       X = OFFSET TO DESTINATION
                   ;
                   ;       JSR     DSTORA
                   ;
                   ;       Y = 'ACC' OFFSET
                   ;       DTAB(X) = 'ACC'
                   ;
B445 A0 12         DSTORA  LDY     #ACC-DTAB
B447 4C 4A B3              JMP     DMOVI


                   ;
                   ; DADDA -- ADD DATA TO 'ACC'
                   ;
                   ; CALLING SEQUENCE:
                   ;
                   ;       Y = OFFSET TO DATA
                   ;
                   ;       JSR     DADDA
                   ;
                   ;       X = 'ACC' OFFSET
                   ;       'ACC' = 'ACC' + DTAB(Y)
                   ;
B44A A2 12         DADDA   LDX     #ACC-DTAB
B44C 4C 55 B3              JMP     DADDI


                   ;
                   ; DSUBA -- SUBTRACT DATA FROM 'ACC'
                   ;
                   ; CALLING SEQUENCE:
                   ;
                   ;       Y = OFFSET TO DATA
                   ;
```

```
                        ;       JSR     DSUBA
                        ;       BEQ     RESULT = 0
                        ;
                        ;       X = 'ACC' OFFSET
                        ;       'ACC' = 'ACC' - DTAB(Y)
                        ;
    B44F A2 12          DSUBA   LDX     #ACC-DTAB
    B451 4C 65 B3               JMP     DSUBI


                        ;
                        ; DCMPA -- COMPARE 'ACC' WITH DATA
                        ;
                        ; CALLING SEQUENCE:
                        ;
                        ;       Y = DATA OFFSET
                        ;
                        ;       JSR     DCMPA
                        ;
                        ;       CC = 'ACC' : DTAB(Y)  (UNSIGNED)
                        ;       X = 'ACC' OFFSET
                        ;
    B454 A2 12          DCMPA   LDX     #ACC-DTAB
    B456 4C 3D B3               JMP     DCMPI

    09FD                TOPV    =V
                        .END
```

```
           SYMBOL TABLE
ABYPAS  B07A      ACC     00A8      ACHR    0001      ACTION  AC46
ADDCOR  030E      ADDR1   A85C      ADDR2   A871      ADDR3   A86E
ADRESS  0064      AFP     D800      ALLPOT  D208      ALPHA   AD3E
ANM     07B1      ANMEND  0004      ANMNDX  0704      ANMRAT  07A9
ANMTBL  A4F4      ANTIC   D400      APPEND  0001      APPMHI  000E
APTR    0090      APWR2   0002      ASCBUF  0706      ATACHR  02FB
ATAN    BE43      ATRACT  004D      AUDC1   D201      AUDC2   D203
AUDC3   D205      AUDC4   D207      AUDCTL  D208      AUDF1   D200
AUDF2   D202      AUDF3   D204      AUDF4   D206      BACPTR  00BC
BADIOC  0086      BADMOD  0091      BB      079C      BETA    AD3B
BFENHI  0035      BFENLO  0034      BITMSK  006E      BLIM    028A
BLKBDV  E471      BLKMOD  A245      BLNKBX  A5C8      BLOKLP  A34A
BOOT?   0009      BOOTAD  0242      BOTSCR  02BF      BOTTOM  0798
BOX     A518      BPTR    003D      BRKABT  0080      BRKKEY  0011
BUFADR  0015      BUFCNT  006B      BUFRFL  0038      BUFRHI  0033
BUFRLO  0032      BUFSTR  006C      BWRTLP  A5EE      BYTCNT  09E9
BYTEND  0792      BYTSTR  078E      BYTWID  0794      CALSEQ  A6A6
CASBUF  03FD      CASETV  E440      CASFLG  030F      CASINI  0002
CASORG  EF41      CASSBT  004B      CASSET  0043      CAUX1   023C
CAUX2   023D      CBAUDH  02EF      CBAUDL  02EE      CBDLP1  A479
CBNDLP  A276      CCOMND  023B      CDEVIC  023A      CDTMA1  0226
CDTMA2  0228      CDTMF3  022A      CDTMF4  022C      CDTMF5  022E
CDTMV1  0218      CDTMV2  021A      CDTMV3  021C      CDTMV4  021E
CDTMV5  0220      CH      02FC      CH1     02F2      CHACT   02F3
CHACTL  D401      CHAR    02FA      CHBAS   02F4      CHBASE  D409
CHKDWN  B1AE      CHKERR  008F      CHKMCH  A80E      CHKRT   B181
CHKSNT  003B      CHKSUM  0031      CHKUP   B196      CHRCNT  0780
CHRORG  E000      CHSPD   078A      CIOCHR  002F      CIOINV  E46E
CIOORG  E4A6      CIOV    E456      CIX     00F2      CKEY    004A
CLOOP2  A759      CLOSE   000C      CLRLUP  A73B      CLRSEQ  A4C2
CLTRNS  09EA      CMPAR   A810      CMPDON  A8B6      CNTEND  0781
COLAC   0072      COLBK   D01A      COLCRS  0055      COLDST  0244
COLDSV  E477      COLINC  007A      COLOR0  02C4      COLOR1  02C5
COLOR2  02C6      COLOR3  02C7      COLOR4  02C8      COLOUR  07B9
COLPF0  D016      COLPF1  D017      COLPF2  D018      COLPF3  D019
COLPM0  D012      COLPM1  D013      COLPM2  D014      COLPM3  D015
COLR    09F0      COLRSH  004F      COMPAR  A8A7      CONSOL  D01F
COS     BD73      COUNTR  007E      CPATTB  AE50      CPTR    0094
CR      009B      CREAD   A54C      CREMOD  A250      CRETRY  0036
CRITIC  0042      CRSINH  02F0      CRSROR  008D      CSOPIV  E47D
CSTART  A001      CSTAT   0288      CTASK   0088      CTIA    D000
CTSTBR  A532      CURSOR  AC13      CURTSK  0088      CUSED   A3EF
CVSPD   078B      DADDA   B44A      DADDI   B355      DADDS   B428
DATLEN  0785      DAUX1   030A      DAUX2   030B      DBSECT  0241
DBUFHI  0305      DBUFLO  0304      DBYTHI  0309      DBYTLO  0308
DCB     0300      DCM090  B349      DCMPA   B454      DCMPI   B33D
DCOMND  0302      DDA010  B435      DDA030  B436      DDA040  B43F
DDEVIC  0300      DDI006  B3D9      DDI008  B3E4      DDI010  B3E5
DDI090  B404      DDI092  B40C      DDI095  B419      DDIVI   B3B6
DEGFLG  00FB      DEGON   0006      DELETE  0021      DELIM   A8CE
DELTAC  0077      DELTAR  0076      DELTAT  0016      DELTAX  0014
DELTAY  0015      DERROR  0090      DFLAGS  0240      DIGRT   00F1
DINDEX  0057      DIRECT  0002      DISK    0044      DISKIV  E450
DISPLY  0053      DLIMCH  072F      DLISTH  D403      DLISTL  D402
DLOADA  B440      DLPTR   00AE      DLPTR2  00B0      DMACTL  D400
DMASK   02A0      DMOVI   B34A      DMU010  B384      DMU020  B39D
DMU090  B3AB      DMULI   B377      DNACK   008B      DNEGI   B41A
DOSINI  000C      DOSVEC  000A      DPTR    0092      DRAWBX  A580
DRAWLN  0011      DRETRY  0037      DRKMSK  004E      DROP    B018
```

```
DROPLP  B02D    DSKFMS  0018    DSKINV  E453    DSKORG  EDEA
DSKTIM  0246    DSKUTL  001A    DSPFLG  02FE    DSTAT   004C
DSTATS  0303    DSTORA  B445    DSUBA   B44F    DSUBI   B365
DTAB    0096    DTIMLO  0306    DUNIT   0301    DUNUSE  0307
DVSTAT  02EA    EDIMOD  A3A6    EDIPTR  00B4    EDITRV  E400
EEXP    00ED    ELECNT  0782    END0    AB12    ENDBOX  A61B
ENDCAL  A736    ENDCHK  A839    ENDCNT  0783    ENDDLM  A8D7
ENDFIL  A8CD    ENDFLD  A895    ENDL1   AA38    ENDL2   AA8C
ENDL3   AAE0    ENDLIN  09F7    ENDPT   0074    ENDRST  AD60
ENDSTR  A9E4    ENDWX   079F    ENDWY   07A0    ENDY    00C6
EOFERR  0088    ERASLP  AFE9    ERRFLG  023F    ERROR   0784
ERSCRS  ACC6    ESCFLG  02A2    ESIGN   00EF    EVBHND  A242
EX      09F9    EXP     DDC0    EXP10   DDCC    FADD    DA66
FASC    D8E6    FBYTWD  078D    FCHRFL  00F0    FCNPTR  008C
FDIV    DB28    FEOF    003F    FIELD   0730    FIGINC  0701
FIGLST  07E1    FIGPTR  0080    FILDAT  02FD    FILFLG  02B7
FILLIN  0012    FILRAM  A737    FINDX   00C3    FLCHK1  A2AD
FLCHK2  A2F5    FLCHK3  A3CE    FLDOP   DD8D    FLDOR   DD89
FLD1P   DD9C    FLD1R   DD98    FLDFIL  A8B7    FLDFL1  A8C0
FLDLEN  0008    FLGPTR  008E    FLNXT   A2B1    FLNXT2  A3D2
FLPTR   00FC    FMOVE   DDB6    FMSZPG  0043    FMUL    DADB
FNCNOT  0092    FORMAT  0022    FPI     D9D2    FPREC   0006
FPSCR   05E6    FPSCR1  05EC    FPTR2   00FE    FR0     00D4
FR1     00E0    FR2     00E6    FRE     00DA    FREQ    0040
FRMERR  008C    FRX     00EC    FSCR    05E6    FSCR1   05EC
FSTOP   DDAB    FSTOR   DDA7    FSUB    DA60    FTYPE   003E
GETASC  A83A    GETCHR  0007    GETFLD  A859    GETKEE  A15A
GETREC  0005    GLBABS  02E0    GPRIOR  026F    GRACTL  D01D
GRAFM   D011    GRAFP0  D00D    GRAFP1  D00E    GRAFP2  D00F
GRAFP3  D010    GTANM1  A32C    GTCH    A812    GTCHR   A8A9
GXMISL  1180    GXPM    1000    HATABS  031A    HDELTA  09FB
HITCLR  D01E    HOLD1   0051    HOLD2   029F    HOLD3   029D
HOLD4   02BC    HOLD5   02BD    HOLDCH  007C    HORM0   0789
HPOSM0  D004    HPOSM1  D005    HPOSM2  D006    HPOSM3  D007
HPOSP0  D000    HPOSP1  D001    HPOSP2  D002    HPOSP3  D003
HSCROL  D404    ICAX1   034A    ICAX1Z  002A    ICAX2   034B
ICAX2Z  002B    ICBAH   0345    ICBAHZ  0025    ICBAL   0344
ICBALZ  0024    ICBLH   0349    ICBLHZ  0029    ICBLL   0348
ICBLLZ  0028    ICCOM   0342    ICCOMT  0017    ICCOMZ  0022
ICDNO   0341    ICDNOZ  0021    ICHID   0340    ICHIDZ  0020
ICIDNO  002E    ICPTH   0347    ICPTHZ  0027    ICPTL   0346
ICPTLZ  0026    ICSPR   034C    ICSPRZ  002C    ICSTA   0343
ICSTAZ  0023    IENTER  B149    IFP     D9AA    INBUFF  00F3
INIT    A000    INIXPS  A63A    INK     A763    INKCLR  0787
INKIT   A7A9    INSCLR  0020    INSDAT  007D    INSLP   A89E
INSRT   A89C    INTABS  0200    INTEMP  022D    INTINV  E46B
INTORG  E6D5    INTZBS  0010    INVFLG  02B6    IOCB    0340
IOCBAS  0020    IOCBSZ  0010    IOCFRE  00FF    IRQEN   D20E
IRQST   D20E    JDON    B1C3    JMPTBL  A1B3    JOFF2   AED1
JOY12   09F8    KBCODE  D209    KBD     004B    KBDORG  F3E4
KEYBDV  E420    KEYBRD  A1E7    KEYDEL  02F1    L1LOOP  A9EF
L2LOOP  AA43    L3LOOP  AA97    LB      0799    LBFEND  05FF
LBPR1   057E    LBPR2   057F    LBUFF   0580    LEDGE   0002
LEN0    09E4    LENCTR  09E8    LENL1   09E1    LENL2   09E2
LENL3   09E3    LINBUF  0247    LINE    09F3    LINELP  AE9C
LINZBS  0000    LMARGN  0052    LOCKFL  0023    LOCOBJ  00BE
LOG     DECD    LOG10   DED1    LOGCOL  0063    LOGMAP  02B2
LOOP0   AAE9    LOWBND  2400    LPENH   0234    LPENV   0235
LPTRTB  A8D8    LSB     09EF    LSTPTR  00AA    M0PF    D000
M0PL    D008    M1PF    D001    M1PL    D009    M2PF    D002
```

```
M2PL    D00A    M3PF    D003    M3PL    D00B    MACHNM  A40D
MAINLP  A12F    MAL100  B243    MAL200  B26E    MAL300  B27C
MALLOC  B201    MAXDEV  0021    MAXIOC  0080    MBC     00A6
MDA100  B2BF    MDA200  B2EA    MDEALL  B27F    MDP     00A4
MEMA    009E    MEMB    00A0    MEMLO   02E7    MEMTOP  02E5
MINIT   B1F2    MLTTMP  0066    MODE    0786    MODELP  A1B0
MODEM   004D    MODFLG  07A7    MODPTR  0084    MONORG  F0E3
MOVDA   B310    MOVIA   B2EB    MPYLP   A897    MPYPWR  A896
MSB     09ED    MSKL1   AA1A    MSKL2   AA6E    MSKL3   AAC2
MSP     00A2    MVCRS   A7E6    MVD010  B326    MVD020  B331
MVD090  B33C    MVI010  B2F4    MVI020  B2FF    MVI030  B306
MVI090  B30F    MXDMOD  0010    N       0000    NAMEND  0008
NAMNDX  0703    NAMTBL  0738    NCHR    0008    NENTRY  A822
NEWCOL  0061    NEWROW  0060    NEWX    0008    NEWY    0009
NEXTOB  B0E3    NMIEN   D40E    NMIRES  D40F    NMIST   D40F
NMNSRT  A316    NOACTN  AC45    NOCKSM  003C    NOEDGE  AC33
NOMTCH  A2E7    NONDEV  0082    NOTE    0026    NOTOPN  0085
NOVAL   A66B    NSB     09EE    NSIGN   00EE    NUSED   A2C9
NVALID  0084    NXTOBJ  0000    OB1     BE60    OB3     BE70
OBJ1    0660    OBJ3    0670    OBJFLG  07BB    OBLIST  09EB
OBNO    07BA    OCHR    0004    OCOLOR  000A    OFF3    A9E1
OFFEND  09F5    OFFSET  09F4    OLDADR  005E    OLDCHR  005D
OLDCOL  005B    OLDLEN  000C    OLDROW  005A    OLDWID  000B
OLDX    0006    OLDY    0007    OPEN    0003    OPNIN   0004
OPNIND  000C    OPNOT   0008    OPNTMP  0066    ORN     07B5
ORNEND  0008    ORNNDX  0705    ORNRAT  07AD    ORNTBL  A4F8
ORPWR2  0003    ORTABL  AB13    OVRRUN  008E    P       B10F
P0PF    D004    P0PL    D00C    P1PF    D005    P1PL    D00D
P2PF    D006    P2PL    D00E    P3PF    D007    P3PL    D00F
PACTL   D302    PADDL0  0270    PADDL1  0271    PADDL2  0272
PADDL3  0273    PADDL4  0274    PADDL5  0275    PADDL6  0276
PADDL7  0277    PAKCOL  09F1    PBCTL   D303    PBPNT   001D
PBUFSZ  001E    PBWID   00C5    PCOLR0  02C0    PCOLR1  02C1
PCOLR2  02C2    PCOLR3  02C3    PENH    D40C    PENV    D40D
PIA     D300    PICADD  0004    PIXLEN  0790    PIXWID  078F
PLACE   AE54    PLYARG  05E0    PLYEVL  DD40    PMBASE  D407
POFF0   AEE6    POFF1   AF05    POFF2   AF4A    POFF3   AF8F
POINT   0025    POKEY   D200    POKMSK  0010    PORTA   D300
PORTB   D301    POT0    D200    POT1    D201    POT2    D202
POT3    D203    POT4    D204    POT5    D205    POT6    D206
POT7    D207    POTGO   D20B    PRELP0  AAE7    PRELP1  A9ED
PRELP2  AA41    PRELP3  AA95    PRINTR  0050    PRINTV  E430
PRIOR   D01B    PRNBUF  03C0    PRNORG  EE78    PRVOPN  0081
PRVPTA  09E5    PRVTRO  07A8    PSAVE   00C4    PSPTR   00C7
PTEMP   001F    PTIMOT  001C    PTINDX  07A5    PTR     00C0
PTRIG0  027C    PTRIG1  027D    PTRIG2  027E    PTRIG3  027F
PTRIG4  0280    PTRIG5  0281    PTRIG6  0282    PTRIG7  0283
PUTCHR  000B    PUTREC  0009    PX00TB  AE4C    PX11TB  AE48
QLINLP  AEE5    QOFFLP  AED4    QRJUS   AE05    QUEATT  ADB7
RADFLG  00FB    RADON   0000    RAMBAC  1400    RAMLO   0004
RAMPTR  00AC    RAMSIZ  02E4    RAMTOP  006A    RANDOM  D20A
RB      079B    RBLOKV  E47A    RDCELL  A640    RDONLY  0087
READPX  ADCE    RECVDN  0039    REDGE   0027    RENAME  0020
RIGHT   0797    RJUSLP  ADFC    RLIST1  07C1    RLIST2  07D1
RMARGN  0053    RMPTR2  00B2    ROWAC   0070    ROWCRS  0054
ROWINC  0079    RPTCNT  09E6    RSTPTR  AD8C    RTO     09E7
RTCL    0700    RTCLOK  0012    RUNMOD  A4A0    S1H     0098
S1L     0096    S2H     009C    S2L     009A    SAVADR  0068
SAVIO   0316    SAVMOD  A285    SAVMSC  0058    SCH010  AD6E
SCH020  ADAD    SCHED   AD61    SCNEND  0050    SCREDT  0045
```

```
SCRENV  E410    SCRFLG  02BB    SCRMEM  0093    SCRPTR  0082
SCRWID  0028    SDLSTH  0231    SDLSTL  0230    SDMCTL  022F
SDSPLY  A1EA    SENDEV  E468    SEG1    B094    SERIN   D20D
SEROUT  D20D    SETVBV  E45C    SHFAMT  006F    SHFLOK  02BE
SHFTO   AAE1    SHFTL1  A9E5    SHFTL2  AA39    SHFTL3  AABD
SIN     BD81    SINDX   00C2    SIOINV  E465    SIOORG  E944
SIOV    E459    SIZE    A61C    SIZEM   D00C    SIZEP0  D008
SIZEP1  D009    SIZEP2  D00A    SIZEP3  D00B    SKCBAC  AC7E
SKCFWD  AC65    SKCLFT  AC97    SKCRIT  ACB0    SKCTL   D20F
SKRES   D20A    SKSTAT  D20F    SOFFND  0791    SOFFST  0793
SOUNDR  0041    SPECIL  000E    SQR     BEB1    SRESTO  ACCC
SRTIMR  022B    SSFLAG  02FF    SSKCTL  0232    STACKP  0318
START   ADA9    STARTY  09F6    STATIS  000D    STATUS  0030
STENDY  AEB0    STICK0  0278    STICK1  0279    STICK2  027A
STICK3  027B    STIMER  D209    STK0    B159    STK1    B14F
STKINI  B10F    STKLP   B125    STKNO   0012    STOCLR  A7E3
STOJOY  B15E    STORE4  A998    STRIG0  0284    STRIG1  0285
STRIG2  0286    STRIG3  0287    STRLST  0086    STRTCH  072E
STWAIT  B1DF    SUBTMP  029E    SUCCES  0001    SWPFLG  007B
SYSVBV  E45F    TABMAP  02A3    TASK1   0600    TASK2   0620
TASK4   0640    TB      079A    TBLPTR  008A    TCBLP   A804
TCOLOR  0013    TCOUNT  0017    TEMP    023E    TEMP1   0312
TEMP2   0314    TEMP3   0315    TIMER1  030C    TIMER2  0310
TIMFLG  0317    TIMOUT  008A    TINDEX  0293    TLOCO   0005
TMP1    07BF    TMP2    07C0    TMPCHR  0050    TMPCOL  02B9
TMPLBT  02A1    TMPNDX  0702    TMPROW  02B8    TMPX1   029C
TNEXT   0000    TOPV    09FD    TRAMSZ  0006    TRETA   0002
TRIG0   D010    TRIG1   D011    TRIG2   D012    TRIG3   D013
TRNRCD  0089    TSCONT  ADA3    TSK1    BE00    TSK2    BE20
TSK4    BE40    TSKSET  A802    TST1    A7CB    TST2    A7D4
TST3    A7DD    TSTAT   0319    TSTDAT  0007    TSTT    0004
TXPOS   0010    TXTCOL  0291    TXTMSC  0294    TXTOLD  0296
TXTROW  0290    TYPOS   0011    UNLOCK  0024    UPDATE  AFD4
UPDFLG  07A6    UPDINI  AFD9    USAREA  0480    V       09FD
VBFLAG  078C    VBHND   A1ED    VBREAK  0206    VCOUNT  D40B
VCTABL  E480    VDELAY  D01C    VDELTA  09FC    VDSLST  0200
VECTBL  E400    VIMIRQ  0216    VINTER  0204    VKEYBD  0208
VPRCED  0202    VRTM0   0788    VSCROL  D405    VSERIN  020A
VSEROC  020E    VSEROR  020C    VTIMR1  0210    VTIMR2  0212
VTIMR4  0214    VVBLKD  0224    VVBLKI  0222    WARMST  0008
WARMSV  E474    WHY     09FA    WMODE   0289    WRITPX  AE09
WRONLY  0083    WRTCRS  ACBA    WRTEMP  09F2    WRTX    079D
WRTY    079E    WSYNC   D40A    XBOX    07A1    XITVBV  E462
XMTDON  003A    XPOS    0795    YBOX    07A3    YDONE   A68D
YPOS    0796    YTEST   A67F    Z       00C9    ZIOCB   0020
ZTEMP1  00F5    ZTEMP3  00F9    ZTEMP4  00F7
```