```
*********************************
*           3D-DEMO             *
*                               *
*      BY MARC GOLOMBECK        *
*                               *
*   VERSION 1.61 / 07.05.2017   *
*                               *
* NEEDS THE FOLLOWING DATA      *
* ALREADY LOADED INTO MEMORY:   *
*                               *
* $7000: SINE/COSINE-TABLE      *
* $7200: POINT AND LINE DATA    *
* $7400: SHAPE TABLE            *
* $7A00: MULTIPLICATION TABLE   *
* $8200: PROJECTION TABLE       *
*                               *
* CONTROLLED VIA JOYSTICK AND   *
* KEYBOARD:                     *
*                               *
* PDL(0/1): ROTATION+ZOOM       *
* PBN(0/1): RESET+STOP/EXIT     *
* KEYS 1-5: CHOOSE 3D-OBJECT    *
* KEYS +/-: ALTERNATE ZOOM      *
* KEY B   : TOGGLE BENCHMARK    *
* KEY Q   : QUIT PROGRAM        *
*                               *
*********************************
*
               ORG   $6000
*
HGR2           EQU   $F3D8      ; SWITCH TO HIRES2
HGR            EQU   $F3E2      ; SWITCH TO HIRES1
HCLR           EQU   $F3F2      ; CLEAR HIRES SCREEN TO BLACK1
HCOLOR         EQU   $F6F0      ; SET HCOLOR
HPOSN          EQU   $F411      ; SET HIRES-CURSOR NO DRAW
HPLOT          EQU   $F457      ; DRAW HIRES-PIXEL
HLIN           EQU   $F53A      ; DRAW HIRES-LINE
PREAD          EQU   $FB1E      ; READ PADDLES
PB0            EQU   $C061      ; PUSH-BUTTON 0
PB1            EQU   $C062      ; PUSH-BUTTON 1
PB2            EQU   $C063      ; PUSH-BUTTON 2
HOME           EQU   $FC58      ; CLEAR SCREEN
COUT           EQU   $FDED      ; PRINT CHARACTER
KYBD           EQU   $C000      ; READ KEYBOARD
STROBE         EQU   $C010      ; CLEAR KEYBOARD
SHNUM          EQU   $F730      ; GET ADRESS OF SHAPE NUMBER
SHDRAW         EQU   $F605      ; DRAW SHAPE ON SCREEN
BELL           EQU   $FBDD      ; RING BELL
WAIT           EQU   $FCA8      ; WAIT A BIT
*
SINTAB         EQU   $7000      ; BASE ADRESS OF SINE TABLE
COSTAB         EQU   $7040      ; BASE ADRESS OF COSINE TABLE
NUMPNT         EQU   $7200      ; NUMBER OF POINTS TO DRAW
TABLE          EQU   $7201      ; BASE ADRESS FOR POINT TABLE
SHTAB          EQU   $7400      ; BASE ADRESS OF SHAPES-68
SSQLO          EQU   $7A00      ; MULT TAB PART 1
SSQHI          EQU   $7C00      ; MULT TAB PART 2
DSQLO          EQU   $7E00      ; MULT TAB PART 3
DSQHI          EQU   $8000      ; MULT TAB PART 4
PROJTAB        EQU   $8200      ; PROJECTION TAB
YLOOKLO        EQU   $8300      ; LOOKUP LINE BASE ADRESS
EVENCOL        EQU   $83C0      ; APPLESOFT CODE EVEN COLORS
ODDCOL         EQU   $83C8      ; APPLESOFT CODE UNEVEN COLORS
XORMASK        EQU   $83D0      ; PIXEL MASK
```

```
ANDMASK         EQU   $83D8       ; PIXEL MASK
LEMASK          EQU   $83DF
RIMASK          EQU   $83E6
YLOOKHI         EQU   $83ED       ; LOOK LINE BASE ADRESS
COLLINE         EQU   $84AD
DIV7HI          EQU   $84D5       ; DIVISION BY 7 TABLE
MOD7HI          EQU   $84ED       ; DIVISION BY 7 TABLE
DIV7LO          EQU   $8505       ; DIVISION BY 7 TABLE
MOD7LO          EQU   $8605       ; DIVISION BY 7 TABLE
*
XTRANS          EQU   $8B         ; X-TRANSLATION CENTER SCREEN
YTRANS          EQU   $5F         ; Y-TRANSLATION
ZTRANS          EQU   $FE         ; MOVE 3D OBJECT AWAY FROM CAMERA
*
TXTPTR          EQU   $06         ; POINTER FOR TEXT OUTPUT
G_PAGE          EQU   $06         ; GRAPHIC PAGE TO DRAW TO
PTR             EQU   $08         ; POINTER FOR DOS COMMAND
PNTCNT          EQU   $09         ; WHICH POINT TO DRAW?
SCALE           EQU   $E7         ; SHAPE SCALE
SHTABPTR        EQU   $E8         ; POINTER TO SHAPE-TABLE
LOOPCNTY        EQU   $FA         ; Y-ROT-COUNTER
ROTSPDY         EQU   $FB         ; Y-ROTATIONAL SPEED
LOOPCNTX        EQU   $FC         ; X-ROT-COUNTER
ROTSPDX         EQU   $FD         ; X-ROTATIONAL SPEED
LOOPCNTZ        EQU   $F4         ; Z-ROT-COUNTER
ROTSPDZ         EQU   $F5         ; Z-ROTATIONAL SPEED
SCALING         EQU   $FF         ; SCALING
*
SINX            EQU   $7B
XREGSAVE        EQU   $7B         ; DOUBLE USE!
COSX            EQU   $7C
SINY            EQU   $7D
COSY            EQU   $7E
SINZ            EQU   $7F
COSZ            EQU   $80
PX1             EQU   $85
PX2             EQU   $87
PX3             EQU   $89
PX4             EQU   $8B
XTO             EQU   $89         ; DOUBLE USE
YTO             EQU   $8B         ; DOUBLE USE
RX              EQU   $8D
RY              EQU   $8F
RZ              EQU   $91
QX              EQU   $93
QY              EQU   $94
XPOS            EQU   $95
YPOS            EQU   $96
ZPOS            EQU   $97
XD              EQU   $98
YD              EQU   $9A
TEMP1           EQU   $EB         ; TEMP STORAGE
TEMP2           EQU   $EC         ; TENP STORAGE
MKAND           EQU   $ED         ; MULTIPLICANT
MATOR           EQU   $EF         ; MULTIPLICATOR
*
XDRAW           EQU   $300        ; X-POSITION FOR DRAW
YDRAW           EQU   $302        ; Y-POSITION FOR DRAW
XDRAWO          EQU   $304        ; OLD POSITION X
YDRAWO          EQU   $306        ; OLD POSITION Y
XDRAW2          EQU   $308        ; UNDRAW CURSOR X
YDRAW2          EQU   $30A        ; UNDRAW CURSOR Y
ASCR            EQU   $30B        ; ACTIVE SCREEN TO DRAW
JOYCNT          EQU   $30C        ; COUNTER FOR JOYSTICK REQUEST
```

```
BENCHM          EQU   $30D       ; BENCHMARK ON?
MAXDRAW         EQU   $30E       ; MAXIMUM LINES TO DRAW
HOLDON          EQU   $30F       ; PAUSE PLOTTING
STEPON          EQU   $30A       ; STEP FUNCTION
*
PSLO            EQU   $D8        ; USING FAC-ADRESS RANGE
PSHI            EQU   $DA        ; FOR POINTER IN MULT-TAB
PDLO            EQU   $DC        ; USING ARG-ADRESS RANGE
PDHI            EQU   $DE        ; FOR POINTER IN MULT-TAB
*
ENTRY           JSR   SETUP      ; SETUP GRAPHIC & VARIABLES
*
TABLESET        JSR   WLCMTXT    ; DRAW TOP & BOTTOM TEXT
*
* READ PADDELS
*
LOOP            INC   JOYCNT     ; MAIN LOOP STARTS HERE
                LDA   HOLDON     ; IS HOLDON ACTIVATED?
                BEQ   EVJOY      ; NO
                LDA   STEPON     ; DO A STEP?
                BNE   DOSTEP     ; YES -> ONE STEP
                JMP   READKBD
DOSTEP          DEC   STEPON     ; RESET STEP-TRIGGER
EVJOY           LDA   JOYCNT     ; EAVLUATE JOYSTICK
                CMP   #$04
                BCC   READ1
                LDA   #$00       ; READ PADDLE EVERY 2 CYCLES
                STA   JOYCNT
*
PREAD0          LDX   #$00       ; VARY Y-ROT SPEED
                JSR   PREAD ; PADDLE (0)
                CPY   #$64
                BCC   INCSPDY    ; DECREASE SPEED
                CPY   #$9B
                BCS   DECSPDY    ; INCREASE SPEED
                BCC   PREAD1     ; 256-STEP CIRCLE LOOP
*
DECSPDY         LDA   ROTSPDY    ; DECREASE ANGLE
                CMP   #$02
                BCC   PREAD1     ; LOOPCNT STILL POSITIVE
                DEC   ROTSPDY
                DEC   ROTSPDY
                JMP   PREAD1     ; JUMP TO NEXT COMMAND
INCSPDY         LDA   ROTSPDY    ; DECREASE WAIT COUNTER
                CMP   #$20
                BCS   PREAD1     ; MAXIMUM 10
                INC   ROTSPDY
                INC   ROTSPDY
*
PREAD1          LDX   #$01       ; VARY X-ROT SPEED
                JSR   PREAD      ; PADDLE (1)
                CPY   #$64
                BCC   DECSPDX    ; DECREASE SPEED
                CPY   #$9B
                BCS   INCSPDX    ; INCREASE SPEED
                BCC   READ1      ; 256-STEP CIRCLE LOOP
*
DECSPDX         LDA   ROTSPDX    ; DECREASE ANGLE
                CMP   #$02
                BCC   READ1      ; LOOPCNT STILL POSITIVE
                DEC   ROTSPDX
                DEC   ROTSPDX
                JMP   READ1      ; JUMP TO NEXT COMMAND
INCSPDX         LDA   ROTSPDX    ; DECREASE WAIT COUNTER
```

```
                CMP   #$20
                BCS   READ1      ; MAXIMUM 10
                INC   ROTSPDX
                INC   ROTSPDX
*
READ1           CLC              ; CALC NEXT STEP
                LDX   #$00
                LDA   LOOPCNTY
                BPL   READ02
                INX              ; INIT FOR BENCHMARK
READ02          ADC   ROTSPDY    ; ADD NEXT STEP
                CLC
                ADC   #$F6       ; ACCU = ACCU -10
                STA   LOOPCNTY
                LDA   LOOPCNTY
                BPL   READ01     ;
                INX
READ01          CPX   #01
                BNE   READ03     ; BENCHMARK LOOP START
                LDY   BENCHM
                BEQ   READ03     ; BENCHMARK OFF
                JSR   BELL       ; REING BELL
*
READ03          CLC              ; X-ROTATION
                LDA   LOOPCNTX
                ADC   ROTSPDX    ; ADD NEXT STEP
                CLC
                ADC   #$F6       ; ACCU = ACCU - 10
                STA   LOOPCNTX
                CLC              ; Z-ROTATION
                LDA   LOOPCNTZ
                ADC   ROTSPDZ
                CLC
                ADC   #$F6
                STA   LOOPCNTZ
*
                JSR   INITPNTS   ; CALC NEW POINT POSITIONS
                JSR   UDRWLNS    ; UNDRAW LINES
                JSR   DRWLNS
*
* READ KEYBOARD FOR LOADING NEW OBJECTS
*
READKBD         LDA   KYBD
                BPL   NOKEY1     ; NO KEY IS PRESSED
KEYPOINT        CMP   #$AE       ; KEY '.' IS PRESSED
                BNE   KEYCOMMA
INCSPDZ         LDA   ROTSPDZ    ; INCREASE Z-ROTATION SPEED
                CMP   #$20
                BCS   NOINCZ     ; MAXIMUM 10
                INC   ROTSPDZ
                INC   ROTSPDZ
NOINCZ          JMP   ENDKEY1
KEYCOMMA        CMP   #$AC       ; KEY "," IS PRESSED
                BNE   KEY1
DECSPDZ         LDA   ROTSPDZ    ; DECREASE ANGLE
                CMP   #$02
                BCC   NODECZ     ; LOOPCNT STILL POSITIVE
                DEC   ROTSPDZ
                DEC   ROTSPDZ
NODECZ          JMP   ENDKEY1
KEY1            CMP   #$B1       ; KEY '1' IS PRESSED
                BNE   KEY2
                JSR   LOAD1      ; LOAD OBJECT 1
                JMP   ENDKEY
```

```
KEY2            CMP   #$B2       ; KEY '2' IS PRESSED
                BNE   KEY3
                JSR   LOAD2
                JMP   ENDKEY
KEY3            CMP   #$B3       ; KEY '3' IS PRESSED
                BNE   KEY4
                JSR   LOAD3
                JMP   ENDKEY
KEY4            CMP   #$B4       ; KEY '4' IS PRESSED
                BNE   KEY5
                JSR   LOAD4
                JMP   ENDKEY
KEY5            CMP   #$B5       ; KEY '5' IS PRESSED
                BNE   KEYPLUS
                JSR   LOAD5
                JMP   ENDKEY
NOKEY1          BPL   NOKEY      ; INTERMEDIATE JUMP
KEYPLUS         CMP   #$AB       ; KEY '+' IS PRESSED
                BNE   KEYMINUS
                LDA   STROBE
                LDA   ZTRANS     ; INCREASE DISTANCE
                CMP   #$30       ; WAS $7F
                BCC   ENDKEY1    ; MAIXMUM 127
                DEC   ZTRANS
                DEC   ZTRANS
                DEC   ZTRANS
                DEC   ZTRANS
                BNE   ENDKEY1
KEYMINUS        CMP   #$AD       ; KEY '-' IS PRESSED
                BNE   KEYQ
                LDA   STROBE
                LDA   ZTRANS     ; DECREASE DISTANCE
                CMP   #$D0       ; WAS $43
                BCS   ENDKEY1    ; LOOPCNT STILL POSITIVE
                INC   ZTRANS
                INC   ZTRANS
                INC   ZTRANS
                INC   ZTRANS
                BNE   ENDKEY1    ; JUMP TO NEXT COMMAND
KEYQ            CMP   #$D1       ; KEY 'Q' IS PRESSED
                BNE   KEYB
                LDA   STROBE
                JMP   END        ; END PROGRAM
KEYB            CMP   #$C2       ; KEY 'B' IS PRESSED
                BNE   KEYS
                LDA   BENCHM     ; TOGGLE BENCHMARK ON/OFF
                BEQ   BENCHON
                DEC   BENCHM     ; TOGGLE OFF
                BEQ   ENDKEY1
BENCHON         INC   BENCHM     ; TOGGLE ON
                BNE   ENDKEY1
KEYS            CMP   #$D3       ; KEY 'S' IS PRESSED
                BNE   KEYH
                LDA   STEPON     ; TOGGLE BENCHMARK ON/OFF
                BEQ   SSTEPON
                DEC   STEPON     ; TOGGLE OFF
                BEQ   ENDKEY1
SSTEPON         INC   STEPON     ; TOGGLE ON
                BNE   ENDKEY1
KEYH            CMP   #$C8       ; KEY 'H' IS PRESSED
                BNE   NOKEY
                LDA   HOLDON     ; TOGGLE BENCHMARK ON/OFF
                BEQ   HHOLDON
                DEC   HOLDON     ; TOGGLE OFF
```

```
                BEQ   ENDKEY1
HHOLDON         INC   HOLDON     ; TOGGLE ON
                BNE   ENDKEY1

ENDKEY          JSR   SETUP
                JSR   WLCMTXT    ; WRITE TOP & BOTTOM TEXT
                JSR   INITPNTS   ; SET INITIAL POINT POSITION
ENDKEY1         LDA   STROBE     ; CLEAR KEYPRESS
*
* CHECK PUSHBUTTONS
*
NOKEY           LDA   PB1        ; PB 1 PRESSED?
                BMI   END        ; EXIT PROGRAM!
PB0CHK          LDA   PB0        ; PB 0 PRESSED?
                BPL   GOLOOP     ; NO -> DO NEXT LOOP
                LDA   #$00       ; RESET ASPECT AND STOP ROTATION
                STA   LOOPCNTX   ; BACK TO INIT POS
                STA   LOOPCNTY
                STA   LOOPCNTZ
                LDA   #$0A
                STA   ROTSPDY    ; STOP ROTATION
                STA   ROTSPDX
                STA   ROTSPDZ
*
GOLOOP          JMP   LOOP       ; START AGAIN
*
END             LDA   #$00       ; END PROGRAM
                STA   $C051      ; SWITCH TO TEXT
                STA   $C052
                STA   $C054
                JSR   HOME       ; CLEAR SCREEN
                JMP   $03D0      ; DOS WARM START NO RTS HERE!
*
********************************
* SETUP GRAPHICS & DATA        *
********************************
*
SETUP           LDA   #$00
                STA   $C050      ; SWITCH ON GRAPHICS
                STA   $C057      ; SWITCH TO HIRES
                STA   BENCHM     ; BENCHMARK OFF
                STA   HOLDON     ; HOLD-FUNCTION DEACTIVATED
                STA   STEPON     ; STEP-FUNCTION DEACTIVATED
*
                LDA   #<SHTAB    ; SET ADRESS FOR SHAPE-TABLE
                STA   SHTABPTR
                LDA   #>SHTAB
                STA   SHTABPTR+1
                LDA   #$01
                STA   SCALE      ; SCALE = 1
*
                LDA   #32
                STA   $E6        ; DRAW ON 1
                STA   $C055      ; SHOW 2
                JSR   HCLR       ; CLEAR HGR
                JSR   DRAWBOX    ; DRAW BOX ON HIRES 1
                LDX   #14
                LDY   #01        ; XPOS = 270
                LDA   #6
                JSR   HPOSN
                LDX   #67
                JSR   SHNUM
                LDA   #00
                JSR   SHDRAW
```

```
*
                LDA   #64
                STA   $E6         ; DRAW ON 2
                STA   $C054
                JSR   HCLR        ; CLEAR HGR2
                JSR   DRAWBOX     ; DRAW BOX ON HIRES 2
                LDX   #14
                LDY   #01         ; XPOS = 270
                LDA   #6
                JSR   HPOSN
                LDX   #67
                JSR   SHNUM
                LDA   #00
                JSR   SHDRAW
                LDX   #11
                LDY   #01         ; XPOS = 267
                LDA   #09
                JSR   HPOSN
                LDX   #69
                JSR   SHNUM
                LDA   #00
                JSR   SHDRAW
*
                LDA   #$03        ; INIT HCOLOR = 3
                JSR   HCOLOR
                LDA   #$0A
                STA   ROTSPDX     ; INIT X-ROT SPEED = 0
                STA   ROTSPDZ     ; INIT Z-ROT SPEED = 0
                LDA   #$0F
                STA   ROTSPDY     ; INIT Y-ROT SPEED = 1
*
                LDA   #$00        ; DEFINED POSITIONS AFTER
                STA   LOOPCNTY    ; STARTING THE ALGO OR
                STA   LOOPCNTZ    ; CHANGING 3D-OBJECTS
                LDA   #$10
                STA   LOOPCNTX    ; CURRENT X-STEP
*
                LDA   #$01
                STA   ASCR        ; DRAW ON HIRES 1
                STA   JOYCNT      ; SET JOYSTICK REQUEST COUNTER
*
                LDA   #$30        ; SCALING
                STA   SCALING
                LDA   #$50
                STA   ZTRANS      ; MOVE OBJECT AWAY FROM CAMERA
*
                LDA   #SSQLO/256 ; SETUP MULT-TAB
                STA   PSLO+1
                LDA   #SSQHI/256
                STA   PSHI+1
                LDA   #DSQLO/256
                STA   PDLO+1
                LDA   #DSQHI/256
                STA   PDHI+1
*
*
                RTS
*
*******************************
* DRAW LINES                  *
*******************************
*
DRWLNS          LDA   #$00
                STA   PNTCNT
```

```
                LDA   ASCR       ; CHECK WHICH SCREEN IS ACTIVE
                CMP   #$01
                BEQ   DRWSCR10
                LDA   #$40       ; SCREEN 2 IS ACTIVE
                BPL   DP1        ; ALWAYS POSITIVE
DRWSCR10 LDA          #$20       ; SCREEN 1 IS ACTIVE
DP1             STA   G_PAGE

GETPNT2         LDA   PNTCNT
                ASL
                ASL
                ASL
                ASL              ; X = CNTR*16
                TAX
                INX              ; SKIP 3 BYTES
                INX
                INX
                LDA   TABLE,X    ; GET XPOS LO-BYTE
                STA   XDRAW
                INX
                LDA   TABLE,X    ; GET XPOS HI-BYTE
                STA   XDRAW+1
                INX
                LDA   TABLE,X    ; GET YPOS
                STA   YDRAW
                INX
                INX              ; SKIP 6 BYTES
                INX
                INX
                INX
                INX
                INX
                LDA   #$04
                STA   MAXDRAW    ; NUMBER OF LINES TO DRAW
                STX   XREGSAVE   ; SAVE X-REG ON STACK
*
* EVALUATE POINTS TO DRAW TO
*
                LDX   XREGSAVE
DRWNXT          LDA   TABLE,X
                BEQ   NXTPNT     ; NOTHING MORE TO DRAW
                TAY              ; SUBTRACT 1 TO GET CORREC
                DEY              ; POINT NUMBER
                TYA
                ASL              ; CALC BASE ADRESS OF POINT TO
                ASL              ; DRAW TO
                ASL
                ASL
                TAY              ; Y = POINNUMBER * 16
                INY              ; SKIP 3 BYTES
                INY
                INY
                LDA   TABLE,Y    ; READ OUT POINTS TO DRAW TO
                STA   XTO
                INY
                LDA   TABLE,Y
                STA   XTO+1
                INY
                LDA   TABLE,Y
                STA   YTO
*
                STX   XREGSAVE
                JSR   LINEDRAW   ; DO FAST LINE DRAW
                LDX   XREGSAVE
```

```
*
                DEC   MAXDRAW    ; MAXIMUM NUMBER OF LINES
                BEQ   NXTPNT     ; REACHED? IF YES NEXT POINT
                INX              ; NEXT POINT
                BNE   DRWNXT
*
NXTPNT          INC   PNTCNT
                LDA   PNTCNT
                CMP   NUMPNT
                BCS   DRWEND     ; ALL DONE
                JMP   GETPNT2
DRWEND          LDA   ASCR       ; DISPLAY DRAW-SCREEN
                CMP   #$02
                BEQ   DISP2      ; SHOW SCREEN 2
                LDA   #64
                STA   $E6        ; DRAW ON 2
                STA   $C054      ; SWITCH TO SCREEN1
                INC   ASCR       ; ASCR = 2
                BNE   DRWEND2
DISP2           STA   $C055      ; SWITCH TO SCREEN2
                LDA   #32
                STA   $E6        ; DRAW ON 1
                DEC   ASCR       ; ASCR = 1
DRWEND2         RTS
*
*
*********************************
* INIT POINT POSITIONS          *
*********************************
*
INITPNTS        LDA   #$00
                STA   PNTCNT
*
* EVAL SIN/COS-TAB
*
                LDX   LOOPCNTX   ; INITIAL POS
                LDA   SINTAB,X   ; GET SINUS FOR X
                STA   SINX
                LDA   COSTAB,X   ; GET COSINUS FOR X
                STA   COSX
                LDX   LOOPCNTY
                LDA   SINTAB,X
                STA   SINY
                LDA   COSTAB,X
                STA   COSY
                LDX   LOOPCNTZ
                LDA   SINTAB,X
                STA   SINZ
                LDA   COSTAB,X
                STA   COSZ
*
* ROTATE POINTS AROUND X AND Y
*
GETPNT          LDA   PNTCNT
                ASL
                ASL
                ASL
                ASL              ; X = CNTR*16
                TAX
                LDA   TABLE,X    ; GET XPOS
                STA   XPOS
                INX
                LDA   TABLE,X    ; GET YPOS
                STA   YPOS
```

```
                INX
                LDA   TABLE,X    ; GET ZPOS
                STA   ZPOS
*
* PERFORM Y-ROTATION
*
STRTROT         LDA   XPOS       ; RX = X * COS(Y) + Z * SIN(Y)
                STA   MATOR      ; X * COS(Y)
                LDA   COSY
                STA   MKAND
                JSR   MULT
                STA   PX1
                STY   PX1+1

                LDA   SINY       ; XPOS STILL IN MATOR!
                STA   MKAND      ; X * SIN(Y)
                JSR   MULT
                STA   PX4
                STY   PX4+1

                LDA   ZPOS       ; Z * SIN(Y)
                STA   MATOR
                LDA   SINY
                STA   MKAND
                JSR   MULT
                STA   PX2
                STY   PX2+1

                LDA   COSY       ; RZ = Z * COS(Y) - X * SIN(Y)
                STA   MKAND      ; ZPOS STILL IN MATOR
                JSR   MULT       ; Z * COS(Y)
                STA   PX3
                STY   PX3+1

                CLC              ; X-CCORD:  ADD
                LDA   PX1
                ADC   PX2
                STA   RX
                LDA   PX1+1
                ADC   PX2+1
                STA   RX+1       ; SAVE HI-BYTE AS XPOS
*
                SEC              ; Z-COORD: SUB
                LDA   PX3
                SBC   PX4
                STA   RZ
                LDA   PX3+1
                SBC   PX4+1
                STA   RZ+1
*
                LDA   YPOS       ; RY = YPOS * 1 (=$7F)
                STA   MATOR
                LDA   #$7F
                STA   MKAND
                JSR   MULT
                STA   RY
                STY   RY+1
*
* PERFORM X-ROTATION
*
                LDA   RX+1       ; RXNEW = RX+1 * 1 (=$7F)
                STA   MATOR
                LDA   #$7F
                STA   MKAND
```

```
                JSR   MULT
                STA   RX
                STY   RX+1
*
                LDA   RY+1        ; RY = Y * COS(X) - Z * SIN(X)
                STA   MATOR       ; Y * COS(X)
                LDA   COSX
                STA   MKAND
                JSR   MULT
                STA   PX1
                STY   PX1+1
*
                LDA   SINX        ; CALC Y * SIN(X) FOR Z-COORD
                STA   MKAND       ; Y * SIN(X)
                JSR   MULT
                STA   PX3
                STY   PX3+1
*
                LDA   RZ+1        ; Z * SIN(X)
                STA   MATOR
                LDA   SINX
                STA   MKAND
                JSR   MULT
                STA   PX2
                STY   PX2+1
*
                LDA   COSX        ; CALC Z * COS(X) FOR Z-CCORD
                STA   MKAND       ; Z * COS(X)
                JSR   MULT
                STA   PX4
                STY   PX4+1
*
                SEC               ; Y-COORD: SUB
                LDA   PX1
                SBC   PX2
                STA   RY
                LDA   PX1+1
                SBC   PX2+1
                STA   RY+1
*
                CLC               ; Z-COORD: ADD + TRANSLATE
                LDA   PX3
                ADC   PX4
                STA   RZ
                LDA   PX3+1
                ADC   PX4+1
                CLC
                ADC   ZTRANS      ; Z-TRANSLATION AND SCALE DOWN
                LSR
                LSR
                STA   RZ+1
*
* PERFORM Z-ROTATION
*
* DO NOT TOUCH RZ ANYMORE!
*
                LDA   RX+1        ; RX = X * COSZ - Y * SINZ
                STA   MATOR       ; X * COS(Z)
                LDA   COSZ
                STA   MKAND
                JSR   MULT
                STA   PX1
                STY   PX1+1
*
```

```
                LDA   SINZ       ; X * SIN(Z)
                STA   MKAND
                JSR   MULT
                STA   PX3
                STY   PX3+1
*
                LDA   RY+1       ; Y * SIN(Z)
                STA   MATOR
                LDA   SINZ
                STA   MKAND
                JSR   MULT
                STA   PX2
                STY   PX2+1
*
                LDA   COSZ       ; Y * COS(Z)
                STA   MKAND
                JSR   MULT
                STA   PX4
                STY   PX4+1
*
                SEC              ; NOW CALCULATE NEW RX
                LDA   PX1        ; SINCE OLD RX IS NEEDED FOR
                SBC   PX2        ; RY-CACLULUS BEFORE!
                STA   RX
                LDA   PX1+1
                SBC   PX2+1
                STA   RX+1
*
                CLC              ; Y-COORD: ADD
                LDA   PX3
                ADC   PX4
                STA   RY
                LDA   PX3+1
                ADC   PX4+1
                STA   RY+1
*
* PERFORM SCALING AND TRANSLATION
*
                ASL   RX
                ROL   RX+1
                ASL   RX
                ROL   RX+1
                ASL   RY
                ROL   RY+1
                ASL   RY
                ROL   RY+1
*
                LDX   RZ+1       ; LOAD PROJECTION
                LDA   PROJTAB,X
                STA   MATOR
                LDA   RX+1       ; ONLY HI-BYTE!
                STA   MKAND
                JSR   UMULT
                STY   QX
*
                LDA   RY+1       ; ONLY HI-BYTE!
                STA   MKAND
                JSR   UMULT
                STY   QY
*
                CLC              ; XD = QX + XT
                LDA   QX
                ADC   #XTRANS
                STA   XD
```

```
                LDA   #$00
                STA   XD+1
                LDA   QX
                BMI   CALCYD     ; IF QX < 0 THEN DO NOT ADD CARRY!
                LDA   #$00
                ADC   #$00       ; ADD CARRY BIT
                STA   XD+1
CALCYD          CLC              ; YD = QY + YT
                LDA   QY
                ADC   #YTRANS
                STA   YD         ; ONLY 1 BYTE HERE, NO CARRY!
*
* WRITE DATA BACK TO TABLE
*
WRTPNT          LDA   PNTCNT
                ASL
                ASL
                ASL
                ASL              ; X = CNTR*16
                TAX
                INX
                INX
                INX              ; SKIP 3 BYTES
                LDA   XD         ; SAVE XDRAW POSITION
                STA   TABLE,X
                INX
                LDA   XD+1
                STA   TABLE,X
                INX
                LDA   YD         ; SAVE YDRAW POSITION
                STA   TABLE,X
*
                INC   PNTCNT
                LDA   PNTCNT
                CMP   NUMPNT
                BCS   INITEND    ; ALL DONE
                JMP   GETPNT
INITEND         RTS
*
                CYC   OFF
*
********************************
* DRAW BOX AROUND HIRES SCREEN *
********************************
*
DRAWBOX         LDX   #$03
                JSR   HCOLOR
                LDA   #$00
                TAY
                TAX
                JSR   HPLOT
                LDA   #23
                LDX   #01
                JSR   HLIN
*
                LDA   #23
                LDX   #01
                LDY   #$BF
                JSR   HLIN
*
                LDA   #$00
                LDX   #$00
                LDY   #$BF
                JSR   HLIN
```

```
*
               LDA   #$00
               TAY
               TAX
               JSR   HLIN
*
               LDY   #00
               LDX   #00
               LDA   #12
               JSR   HPOSN
*
               LDA   #23
               LDX   #01
               LDY   #12
               JSR   HLIN
               RTS
*
*********************************
*SIGNED  DIVISION 16 BIT/8 BIT *
*********************************
*
               CYC
*
DIVI           STY   DEND       ;
               STA   DEND+1
               STX   DOR
*
               LDX   #$00
DECHK          LDA   DEND+1     ; DIVIDEND NEG?
               BPL   DORCHK
               INX              ; INC X-REG FOR NEG SIGN
               LDA   DEND
               SEC              ; TWO'S COMPLEMENT
               SBC   #$01
               EOR   #$FF
               STA   DEND
               LDA   DEND+1
               SBC   #$00
               EOR   #$FF
               STA   DEND+1
DORCHK         LDA   DOR        ; DIVISOR NEG?
               BPL   DIVIGO
               INX              ; INC X-REG FOR NEG SIGN
               SEC              ; TWO'S COMPLEMENT
               SBC   #$01
               EOR   #$FF
               STA   DOR
*
DIVIGO         LDA   DEND+1     ; TOO LARGE OR ZERO?
               CMP   DOR        ; CMP HI-BYTE WITH DOR!
               BCC   DLOOP      ; NO -> NO ERROR!
               JMP   DIVERR     ; YES -> ERROR!
DLOOP          ASL   DEND       ; DOUBLE SHIFT DIVIDEND
               ROL              ; DEND+1 STILL IN ACCU!
               BCS   DSUBTR     ; SUBTRACTION WHEN CARRY IS SET
               CMP   DOR
               BCC   DLOOP2
DSUBTR         SBC   DOR
               INC   DEND
DLOOP2         ASL   DEND
               ROL
               BCS   DSUBTR2
               CMP   DOR
               BCC   DLOOP3
```

```
DSUBTR2         SBC   DOR
                INC   DEND
DLOOP3          ASL   DEND
                ROL
                BCS   DSUBTR3
                CMP   DOR
                BCC   DLOOP4
DSUBTR3         SBC   DOR
                INC   DEND
DLOOP4          ASL   DEND
                ROL
                BCS   DSUBTR4
                CMP   DOR
                BCC   DLOOP5
DSUBTR4         SBC   DOR
                INC   DEND
DLOOP5          ASL   DEND
                ROL
                BCS   DSUBTR5
                CMP   DOR
                BCC   DLOOP6
DSUBTR5         SBC   DOR
                INC   DEND
DLOOP6          ASL   DEND
                ROL
                BCS   DSUBTR6
                CMP   DOR
                BCC   DLOOP7
DSUBTR6         SBC   DOR
                INC   DEND
DLOOP7          ASL   DEND
                ROL
                BCS   DSUBTR7
                CMP   DOR
                BCC   DLOOP8
DSUBTR7         SBC   DOR
                INC   DEND
DLOOP8          ASL   DEND
                ROL
                BCS   DSUBTR8
                CMP   DOR
                BCC   DCONT
DSUBTR8         SBC   DOR
                INC   DEND

*
* STORE RESULTS
*
DCONT           STA   DOR         ; MOVE REMAINDER IN DOR
                CLC               ; NO ERROR -> CLEAR CARRY
                CPX   #$01        ; NEG SIGN FOR RESULT?
                BNE   DIVEND      ; NO, SIGN IS POSITIVE -> END
                EOR   #$FF        ; TWO'S COMPLEMENT  OF ACCU
                CLC
                ADC   #$01
                STA   DOR
                LDA   DEND
                EOR   #$FF
                ADC   #$01
                STA   DEND
                LDA   DOR         ; DOR = REMAINDER
DIVEND          LDY   DEND        ; DEND = QUOTIENT
                BNE   DIVRTS
                INY
```

```
DIVRTS          RTS
DIVERR          LDY   #$00       ; RETURN 0 AS RESULT
                RTS
*
                CYC   OFF
*
DEND            DS    2
DOR             DS    1
*
********************************
* 8-BIT UNSIGNED FAST MULTIPLY *
********************************
*
                CYC
*
UMULT           LDY   MKAND
                STY   TEMP1      ; SAVE FOR LATER USE
UMTCHK          LDA   MATOR
                STA   TEMP2      ; SAVE FOR LATER USE
                STA   PSHI
                EOR   #$FF
                STA   PDHI
                SEC
                LDA   (PSHI),Y   ; GET (A+Y)^2/4 (HI-BYTE)
                SBC   (PDHI),Y   ; SUBTRACT (-A+Y)^2/4 (HI-BYTE)
*
* STORE RESULTS
*
                LDY   TEMP1      ; CHECK IF MKAND < 0
                BPL   UMDONE     ; NO -> ALL DONE
                SEC              ; MATOR STILL IN ACCU!
                SBC   TEMP2      ; SUBTRACT TEMP2 FROM MATOR
*
UMDONE          TAY              ; MOVE ACCU TO Y-REG AS RESULT
                RTS
*
                CYC   OFF
*
*
********************************
* 8-BIT SIGNED COMPL MULTIPLY  *
********************************
*
                CYC
*
MULT            LDY   MKAND
                STY   TEMP1      ; SAVE FOR LATER USE
MTCHK           LDA   MATOR
                STA   TEMP2      ; SAVE FOR LATER USE
                STA   PSLO       ; INDEX INTO SUM TABLE BY A
                STA   PSHI
                EOR   #$FF
                STA   PDLO       ; INDEX INTO DIFF TABLE BY -A-1
                STA   PDHI
                LDA   (PSLO),Y   ; GET (A+Y)^2/4 (LO BYTE)
                SEC
                SBC   (PDLO),Y   ; SUBTRACT (-A+Y)^2/4 (LO BYTE)
                STA   MKAND      ; SAVE IT
                LDA   (PSHI),Y   ; GET (A+Y)^2/4 (HI-BYTE)
                SBC   (PDHI),Y   ; SUBTRACT (-A+Y)^2/4 (HI-BYTE)
*
* STORE RESULTS
*
                LDY   TEMP1      ; CHECK IF MKAND < 0
```

```
                BPL   CHKT2      ; NO CHECK MATOR
                SEC              ; MATOR STILL IN ACCU!
                SBC   TEMP2      ; SUBTRACT TEMP2 FROM MATOR
*
CHKT2           LDY   TEMP2      ; CHECK IF MATOR < 0
                BPL   MDONE      ; NO, RTS
                SEC              ; MATOR STILL IN ACCU
                SBC   TEMP1      ; SUBTRACT TEMP1 FROM MATOR
*
MDONE           TAY              ; MOVE ACCU TO Y-REG AS RESULT
                LDA   MKAND      ; LOAD LO BYTE
                RTS
*
                CYC   OFF
*
*********************************
* 8-BIT SIGNED FAST MULTIPLY   *
*********************************
*
                CYC
*
FMULT           LDY   MKAND
                STY   TEMP1      ; SAVE FOR LATER USE
FMTCHK          LDA   MATOR
                STA   TEMP2      ; SAVE FOR LATER USE
                STA   PSHI
                EOR   #$FF
                STA   PDHI
                SEC
                LDA   (PSHI),Y   ; GET (A+Y)^2/4 (HI-BYTE)
                SBC   (PDHI),Y   ; SUBTRACT (-A+Y)^2/4 (HI-BYTE)
*
* STORE RESULTS
*
                LDY   TEMP1      ; CHECK IF MKAND < 0
                BPL   FCHKT2     ; NO CHECK MATOR
                SEC              ; MATOR STILL IN ACCU!
                SBC   TEMP2      ; SUBTRACT TEMP2 FROM MATOR
*
FCHKT2          LDY   TEMP2      ; CHECK IF MATOR < 0
                BPL   FMDONE     ; NO, RTS
                SEC              ; MATOR STILL IN ACCU
                SBC   TEMP1      ; SUBTRACT TEMP1 FROM MATOR
*
FMDONE          TAY              ; MOVE ACCU TO Y-REG AS RESULT
                RTS
*
                CYC   OFF
*
*********************************
* TOP & BOTTOM TEXT MESSAGE    *
*********************************
*
* PRINT OUT WELCOME TEXT AS SHAPES ON SCREEN
*
WLCMTXT         LDX   #$03
                JSR   HCOLOR
                STX   $F9        ; SET ROT = 0
                LDA   #$01
                STA   SCALE      ; SET SCALE = 1
                LDX   #$00
TXTLOOP         LDA   TXTDAT,X
                BEQ   TLP2       ; STOP WHEN $00 IS READ
                STA   SHPDUMP    ; SAVE SHAPE-NUMBER
```

```
                STX   XREGDUMP   ; SAVE X-REG COUNTER STATUS
                TXA              ; MOVE CHAR POSITION TO ACCU
                ASL              ; CHAR-POS * 8 PIXEL
                ASL              ; EACH CHAR NEEDS 8 PIXEL
                ASL              ; SPACING
                CLC
                ADC   #26        ; ADD X-OFFSET
                TAX              ; MOVE X-POS TO X-REG
                STX   SHXPOS     ; SAVE SHAPE X-POS
                LDY   #$00       ; HIGH-BYTE = 0
                LDA   #09        ; Y-OFFSET FOR TEXT OUTPUT
                JSR   HPOSN      ; POSITION THE CURSOR
*
                LDX   SHPDUMP    ; RETRIEVE SHAPE-NUMBER
                JSR   SHNUM      ; GET SHAPE ADRESS
                LDA   #32        ; DRAW ON SCREEN 1
                STA   $E6
                LDA   #$00       ; SET ROT = 0
                JSR   SHDRAW     ; DRAW SHAPE
*
                LDX   SHXPOS     ; GET SHAPE X-POS
                LDY   #$00       ; HIGH-BYTE = 0
                LDA   #09        ; Y-OFFSET FOR TEXT OUTPUT
                JSR   HPOSN      ; POSITION THE CURSOR
                LDX   SHPDUMP    ; RETRIEVE SHAPE-NUMBER
                JSR   SHNUM      ; GET SHAPE ADRESS
                LDA   #64        ; DRAW ON SCREEN 2
                STA   $E6
                LDA   #$00       ; SET ROT = 0
                JSR   SHDRAW     ; DRAW SHAPE
*
                LDX   XREGDUMP
                INX
                BNE   TXTLOOP
*
TLP2            LDX   #$00
TXTLOOP2        LDA   TXTDAT2,X
                BEQ   SUDONE
                STA   SHPDUMP
                STX   XREGDUMP
                TXA
                ASL
                ASL
                ASL
                CLC
                ADC   #18
                TAX
                STX   SHXPOS
                LDY   #$00
                LDA   #188
                JSR   HPOSN
*
                LDX   SHPDUMP
                JSR   SHNUM
                LDA   #32
                STA   $E6
                LDA   #$00
                JSR   SHDRAW
*
                LDX   SHXPOS
                LDY   #$00
                LDA   #188
                JSR   HPOSN
                LDX   SHPDUMP
```

```
                JSR   SHNUM
                LDA   #64
                STA   $E6
                LDA   #$00
                JSR   SHDRAW
                LDX   XREGDUMP
                INX
                BNE   TXTLOOP2
SUDONE          RTS
*
* TEXT MESSAGES WHEN LOADING 3D-OBJECTS
*
TXTOB1          LDA   #<TXTDAT3  ; SET ADRESS FOR MESSAGE
                STA   TXTPTR
                LDA   #>TXTDAT3
                STA   TXTPTR+1
                JSR   DLTTXT
                JMP   PRTTXT     ; PRINT MESSAGE
*
TXTOB2          LDA   #<TXTDAT4
                STA   TXTPTR
                LDA   #>TXTDAT4
                STA   TXTPTR+1
                JSR   DLTTXT
                JMP   PRTTXT
*
TXTOB3          LDA   #<TXTDAT5
                STA   TXTPTR
                LDA   #>TXTDAT5
                STA   TXTPTR+1
                JSR   DLTTXT
                JMP   PRTTXT
*
TXTOB4          LDA   #<TXTDAT6
                STA   TXTPTR
                LDA   #>TXTDAT6
                STA   TXTPTR+1
                JSR   DLTTXT
                JMP   PRTTXT
*
TXTOB5          LDA   #<TXTDAT7
                STA   TXTPTR
                LDA   #>TXTDAT7
                STA   TXTPTR+1
                JSR   DLTTXT
                JMP   PRTTXT
*
PRTTXT          LDY   #$00
                LDX   #$03
                JSR   HCOLOR     ; SET HCOLOR = 3
TXTLOOP3        LDA   (TXTPTR),Y
                BEQ   TXTDONE
                STA   SHPDUMP
                STY   YREGDUMP
                TYA
                ASL
                ASL
                ASL
                CLC
                ADC   #18
                TAY
                TAX
                STY   SHXPOS
                LDY   #$00
```

```
                LDA   #188
                JSR   HPOSN
*
                LDX   SHPDUMP
                JSR   SHNUM
                LDA   #32
                STA   $E6
                LDA   #$00
                JSR   SHDRAW
*
                LDX   SHXPOS
                LDY   #$00
                LDA   #188
                JSR   HPOSN
                LDX   SHPDUMP
                JSR   SHNUM
                LDA   #64
                STA   $E6
                LDA   #$00
                JSR   SHDRAW
*
                LDY   YREGDUMP
                INY
                BNE   TXTLOOP3
TXTDONE         RTS
*
TXTDAT          HEX   292231313A01151135
                HEX   2901232A33352925223A
                HEX   012231312D26013E3C
                HEX   00
*
TXTDAT2         HEX   450114250E25262E3001
                HEX   0E012E0F28302D30
                HEX   2E2326242C01131112
                HEX   18014500
*
TXTDAT3         HEX   2D3022252A2F28013529
                HEX   2601352635332226252633
                HEX   0F0F0F010101010101
                HEX   010100
*
TXTDAT4         HEX   2D3022252A2F28013529
                HEX   2601362E2333262D2D22
                HEX   0F0F0F010101010101
                HEX   010100
*
TXTDAT5         HEX   2D3022252A2F28013529
                HEX   2601342A2E312D260124362326
                HEX   0F0F0F010101010101
                HEX   010100
*
TXTDAT6         HEX   2D3022252A2F28013529
                HEX   2601352635332224362326
                HEX   0F0F0F010101010101
                HEX   010100
*
TXTDAT7         HEX   2D3022252A2F28013529
                HEX   26012436232601353 82A2F34
                HEX   0F0F0F010101010101
                HEX   010100
*
LCNT            DS    1
XREGDUMP        DS    1
YREGDUMP        DS    1
```

```
SHPDUMP         DS    1
SHXPOS          DS    1
*
* DELETE BOTTOM TEXT
*
DLTTXT          LDX   #$00
                JSR   HCOLOR      ; HCOLOR = 0
                LDA   ASCR
DLTSTRT         LDA   #180
                STA   YDELROW     ; Y OF LINE TO DELETE
DLTLOOP         LDA   #32
                STA   $E6         ; DELETE ON SCREEN 1
                LDX   #$02
                LDY   #$00
                LDA   YDELROW
                JSR   HPOSN
                LDA   #$15
                LDX   #$01
                LDY   YDELROW
                JSR   HLIN
                LDA   #64
                STA   $E6         ; DELETE ON SCREEN 2
                LDX   #$02
                LDY   #$00
                LDA   YDELROW
                JSR   HPOSN
                LDA   #$15
                LDX   #$01
                LDY   YDELROW
                JSR   HLIN
                INC   YDELROW
                LDA   YDELROW
                CMP   #189        ; LINE 189 REACHED?
                BCC   DLTLOOP     ; NO -> DELETE NEXT LINE
DLTEND          RTS               ; END OF LOOP
*
YDELROW         DS    1
*
********************************
* LOAD 3D-OBJECTS IN MEMORY    *
********************************
*
LOAD1           JSR   TXTOB1      ; PRINT LOADING MESSAGE
LOAD01          LDA   #$8D        ; LOAD 3D-OBJECT 1 INTO MEMORY
                JSR   COUT
                JSR   PRINT
                HEX   84
                ASC   "BLOAD OBJECT1.3D,A$7200"
                HEX   8D00
                RTS
*
LOAD2           JSR   TXTOB2
LOAD02          LDA   #$8D        ; LOAD 3D-OBJECT 1 INTO MEMORY
                JSR   COUT
                JSR   PRINT
                HEX   84
                ASC   "BLOAD OBJECT2.3D,A$7200"
                HEX   8D00
                RTS
*
LOAD3           JSR   TXTOB3
LOAD03          LDA   #$8D        ; LOAD 3D-OBJECT 1 INTO MEMORY
                JSR   COUT
                JSR   PRINT
```

```
                HEX   84
                ASC   "BLOAD OBJECT3.3D,A$7200"
                HEX   8D00
                RTS
*
LOAD4           JSR   TXTOB4
LOAD04          LDA   #$8D       ; LOAD 3D-OBJECT 1 INTO MEMORY
                JSR   COUT
                JSR   PRINT
                HEX   84
                ASC   "BLOAD OBJECT4.3D,A$7200"
                HEX   8D00
                RTS
*
LOAD5           JSR   TXTOB5
LOAD05          LDA   #$8D       ; LOAD 3D-OBJECT 1 INTO MEMORY
                JSR   COUT
                JSR   PRINT
                HEX   84
                ASC   "BLOAD OBJECT5.3D,A$7200"
                HEX   8D00
                RTS
*
PRINT           PLA
                STA   PTR
                PLA
                STA   PTR+1
                LDY   #$01
P0              LDA   (PTR),Y
                BEQ   PFIN
                JSR   COUT
                INY
                BNE   P0
                CHK
*
PFIN            CLC
                TYA
                ADC   PTR
                STA   PTR
                LDA   PTR+1
                ADC   #$00
                PHA
                LDA   PTR
                PHA
PEXIT           RTS
*
********************************
* UNDRAW LINES                 *
********************************
*
UDRWLNS         LDA   #$00
*
                LDY   $E6
                CPY   #$20
                BEQ   UDRWSCR1
                JMP   UDRWSCR2
UDRWSCR1        LDX   #$0B
UDRWLP1         STA   $2200,X
                STA   $2600,X
                STA   $2A00,X
                STA   $2E00,X
                STA   $3200,X
                STA   $3600,X
                STA   $3A00,X
```

```
STA   $3E00,X

STA   $2280,X
STA   $2680,X
STA   $2A80,X
STA   $2E80,X
STA   $3280,X
STA   $3680,X
STA   $3A80,X
STA   $3E80,X

STA   $2300,X
STA   $2700,X
STA   $2B00,X
STA   $2F00,X
STA   $3300,X
STA   $3700,X
STA   $3B00,X
STA   $3F00,X

STA   $2380,X
STA   $2780,X
STA   $2B80,X
STA   $2F80,X
STA   $3380,X
STA   $3780,X
STA   $3B80,X
STA   $3F80,X

STA   $2028,X
STA   $2428,X
STA   $2828,X
STA   $2C28,X
STA   $3028,X
STA   $3428,X
STA   $3828,X
STA   $3C28,X

STA   $20A8,X
STA   $24A8,X
STA   $28A8,X
STA   $2CA8,X
STA   $30A8,X
STA   $34A8,X
STA   $38A8,X
STA   $3CA8,X

STA   $2128,X
STA   $2528,X
STA   $2928,X
STA   $2D28,X
STA   $3128,X
STA   $3528,X
STA   $3928,X
STA   $3D28,X

STA   $21A8,X
STA   $25A8,X
STA   $29A8,X
STA   $2DA8,X
STA   $31A8,X
STA   $35A8,X
STA   $39A8,X
STA   $3DA8,X
```

```
                STA   $2228,X
                STA   $2628,X
                STA   $2A28,X
                STA   $2E28,X
                STA   $3228,X
                STA   $3628,X
                STA   $3A28,X
                STA   $3E28,X

                STA   $22A8,X
                STA   $26A8,X
                STA   $2AA8,X
                STA   $2EA8,X
                STA   $32A8,X
                STA   $36A8,X
                STA   $3AA8,X
                STA   $3EA8,X

                STA   $2328,X
                STA   $2728,X
                STA   $2B28,X
                STA   $2F28,X
                STA   $3328,X
                STA   $3728,X
                STA   $3B28,X
                STA   $3F28,X

                STA   $23A8,X
                STA   $27A8,X
                STA   $2BA8,X
                STA   $2FA8,X
                STA   $33A8,X
                STA   $37A8,X
                STA   $3BA8,X
                STA   $3FA8,X
*
                STA   $2050,X
                STA   $2450,X
                STA   $2850,X
                STA   $2C50,X
                STA   $3050,X
                STA   $3450,X
                STA   $3850,X
                STA   $3C50,X

                STA   $20D0,X
                STA   $24D0,X
                STA   $28D0,X
                STA   $2CD0,X
                STA   $30D0,X
                STA   $34D0,X
                STA   $38D0,X
                STA   $3CD0,X

                STA   $2150,X
                STA   $2550,X
                STA   $2950,X
                STA   $2D50,X
                STA   $3150,X
                STA   $3550,X
                STA   $3950,X
                STA   $3D50,X
```

```
                STA   $21D0,X
                STA   $25D0,X
                STA   $29D0,X
                STA   $2DD0,X
                STA   $31D0,X
                STA   $35D0,X
                STA   $39D0,X
                STA   $3DD0,X

                INX
                CPX   #29
                BEQ   UDRW1RTS
                JMP   UDRWLP1
UDRW1RTS        RTS

UDRWSCR2        LDX   #$0B
UDRWLP2         STA   $4200,X
                STA   $4600,X
                STA   $4A00,X
                STA   $4E00,X
                STA   $5200,X
                STA   $5600,X
                STA   $5A00,X
                STA   $5E00,X

                STA   $4280,X
                STA   $4680,X
                STA   $4A80,X
                STA   $4E80,X
                STA   $5280,X
                STA   $5680,X
                STA   $5A80,X
                STA   $5E80,X

                STA   $4300,X
                STA   $4700,X
                STA   $4B00,X
                STA   $4F00,X
                STA   $5300,X
                STA   $5700,X
                STA   $5B00,X
                STA   $5F00,X

                STA   $4380,X
                STA   $4780,X
                STA   $4B80,X
                STA   $4F80,X
                STA   $5380,X
                STA   $5780,X
                STA   $5B80,X
                STA   $5F80,X

                STA   $4028,X
                STA   $4428,X
                STA   $4828,X
                STA   $4C28,X
                STA   $5028,X
                STA   $5428,X
                STA   $5828,X
                STA   $5C28,X

                STA   $40A8,X
                STA   $44A8,X
                STA   $48A8,X
```

```
                STA   $4CA8,X
                STA   $50A8,X
                STA   $54A8,X
                STA   $58A8,X
                STA   $5CA8,X
*
                STA   $4128,X
                STA   $4528,X
                STA   $4928,X
                STA   $4D28,X
                STA   $5128,X
                STA   $5528,X
                STA   $5928,X
                STA   $5D28,X

                STA   $41A8,X
                STA   $45A8,X
                STA   $49A8,X
                STA   $4DA8,X
                STA   $51A8,X
                STA   $55A8,X
                STA   $59A8,X
                STA   $5DA8,X

                STA   $4228,X
                STA   $4628,X
                STA   $4A28,X
                STA   $4E28,X
                STA   $5228,X
                STA   $5628,X
                STA   $5A28,X
                STA   $5E28,X

                STA   $42A8,X
                STA   $46A8,X
                STA   $4AA8,X
                STA   $4EA8,X
                STA   $52A8,X
                STA   $56A8,X
                STA   $5AA8,X
                STA   $5EA8,X

                STA   $4328,X
                STA   $4728,X
                STA   $4B28,X
                STA   $4F28,X
                STA   $5328,X
                STA   $5728,X
                STA   $5B28,X
                STA   $5F28,X

                STA   $43A8,X
                STA   $47A8,X
                STA   $4BA8,X
                STA   $4FA8,X
                STA   $53A8,X
                STA   $57A8,X
                STA   $5BA8,X
                STA   $5FA8,X
*
                STA   $4050,X
                STA   $4450,X
                STA   $4850,X
                STA   $4C50,X
```

```
                STA   $5050,X
                STA   $5450,X
                STA   $5850,X
                STA   $5C50,X

                STA   $40D0,X
                STA   $44D0,X
                STA   $48D0,X
                STA   $4CD0,X
                STA   $50D0,X
                STA   $54D0,X
                STA   $58D0,X
                STA   $5CD0,X

                STA   $4150,X
                STA   $4550,X
                STA   $4950,X
                STA   $4D50,X
                STA   $5150,X
                STA   $5550,X
                STA   $5950,X
                STA   $5D50,X

                STA   $41D0,X
                STA   $45D0,X
                STA   $49D0,X
                STA   $4DD0,X
                STA   $51D0,X
                STA   $55D0,X
                STA   $59D0,X
                STA   $5DD0,X

                INX
                CPX   #29
                BEQ   UDRW2RTS
                JMP   UDRWLP2
UDRW2RTS        RTS
*
*********************************
* FAST LINE DRAW                *
*********************************
*
*
HBASL           EQU   $07        ; DOUBLE ZPAGE ADRESS USE!!!
XPOSL           EQU   $7C
XPOSH           EQU   $7D
YPOSN           EQU   $7E
DELTAXL         EQU   $7F
DELTAXH         EQU   $80
DELTAY          EQU   $85
COUNT           EQU   $86
COUNTH          EQU   $87
DIFF            EQU   $88
DIFFH           EQU   $8D
ANDMSK          EQU   $8E
WIDEFLAG        EQU   $8F
*
LINEDRAW        LDA   XTO        ; WHICH X-VALUE IS ON THE LEFT?
                SEC
                SBC   XDRAW
                TAX
                BEQ   CHKVERT    ; LOW BYTE EVEN CHECK VERTICAL
                LDA   XTO+1
                SBC   XDRAW+1
```

```
                BCS   LX0LEFT
*
LX0RIGHT        EOR   #$FF        ; INVERT HI-BYTE
                STA   DELTAXH     ; STORE HI-BYTE
                TXA
                EOR   #$FF        ; INVERT LO-BYTE
                STA   DELTAXL
                INC   DELTAXL     ; TWO'S COMPLEMENT
                BNE   NOINCHI     ; ROLLED INTO HI-BYTE?
                INC   DELTAXH     ; YES
NOINCHI         LDA   XTO         ; START WITH XTO
                STA   XPOSL
                LDA   XTO+1
                STA   XPOSH
                LDA   YTO
                STA   YPOSN
                SEC
                SBC   YDRAW       ; DELTA-Y?
                JMP   LNCOMMON
*
CHKVERT         LDA   XTO+1       ; DIFF IN HI-BYTES?
                SBC   XDRAW+1     ; CARRY STILL SET
                BLT   LX0RIGHT    ; WIDTH = 256, XDRAW RIGHT
                BNE   LX0LEFT     ; WIDTH = 256, XDRAW LEFT
                JMP   VERTLINE
*
LX0LEFT         STX   DELTAXL     ; XDRAW IS LEFT
                STA   DELTAXH
                LDA   XDRAW       ; START WITH XDRAW
                STA   XPOSL
                LDA   XDRAW+1
                STA   XPOSH
                LDA   YDRAW
                STA   YPOSN       ; GET DELTA-Y
                SEC
                SBC   YTO
*
LNCOMMON        BCS   LPOSY
                EOR   #$FF        ; INVERT IF NEGATIVE
                ADC   #$01
                STA   DELTAY
                LDA   #$E8        ; INX
                BNE   GOTDY
LPOSY           STA   DELTAY
                LDA   #$CA        ; DEX
GOTDY           STA   LHMODY      ; SELF-MODIFYING CODE
                STA   LVMODY
                STA   LWMODY
*
* CHECK DOMINANCE
*
                LDY   #$01
                LDA   DELTAXL
                LDX   DELTAXH
                BNE   HORIDOM     ; WIDTH > 256
                CMP   #$FF        ; WIDTH = 255
                BEQ   HORIDOM
                DEY               ; NOT WIDE
                CMP   DELTAY
                BGE   HORIDOM     ; IF DIAGONAL
                JMP   VERTDOM
*
HORIDOM         STY   WIDEFLAG    ; HORIZONTAL IS DOMINANT
                STA   COUNT       ; COUNT = DELTAX + 1
```

```
                INC   COUNT
                LSR             ; DIFF = DELTAX / 2
                STA   DIFF
*
                LDX   XPOSL
                LDA   XPOSH     ; >= 256?
                BEQ   LOTABLE   ; NO -> USE LO-TABLE
                LDY   DIV7HI,X
                LDA   MOD7HI,X
                BPL   GOTTAB    ; BRANCH ALWAYS
LOTABLE         LDY   DIV7LO,X
                LDA   MOD7LO,X
GOTTAB          TAX
                LDA   ANDMASK,X
                STA   ANDMSK
*
                LDX   YPOSN
                LDA   YLOOKLO,X
                STA   HBASL
                LDA   YLOOKHI,X
                ORA   G_PAGE
                STA   HBASL+1
                LDA   WIDEFLAG
                BEQ   NOTWIDE
                JMP   WIDEDOM
NOTWIDE         JMP   HORZLOOP
HRTS            RTS
*
HNOROLL         STA   DIFF      ; LOOP BOTTOM
HDECC           DEC   COUNT
                BEQ   HRTS
*
HORZLOOP        LDA   (HBASL),Y ; PLOT PIXEL
                ORA   ANDMSK    ; SET ADDITIONAL POINTS
                STA   (HBASL),Y ; STORE POINTS
*
                LDA   ANDMSK    ; MOVE RIGHT
                ASL             ; SHIFT & LOSE HI-BIT
                EOR   #$80      ; SET HI-BIT
                BNE   NOH8      ; IF HI-BIT IS CLEARED
                INY             ; NEXT BYTE
                LDA   #$81      ; RESET
NOH8            STA   ANDMSK
*
                LDA   DIFF      ; UPDATE ERROR DIFFERENCE
                SEC
                SBC   DELTAY
                BCS   HNOROLL
                ADC   DELTAXL
                STA   DIFF
LHMODY          INX             ; MODIFYABLE
                LDA   YLOOKLO,X
                STA   HBASL
                LDA   YLOOKHI,X
                ORA   G_PAGE    ; ACTIVE PAGE
                STA   HBASL+1
                BNE   HDECC
*
VERTDOM         LDX   YDRAW
                CPX   YPOSN
                BNE   ENDY0
                LDX   YTO
ENDY0           STX   LVCHK+1
                LDA   DELTAY
```

```
                LSR
                STA   DIFF
*
                LDX   XPOSL
                LDA   XPOSH
                BEQ   LOTABLL
                LDY   DIV7HI,X
                LDA   MOD7HI,X
                BPL   GOTTAB2
LOTABLL         LDY   DIV7LO,X
                LDA   MOD7LO,X
GOTTAB2         TAX
                LDA   ANDMASK,X
                STA   ANDMSK
                LDX   YPOSN
                JMP   VERTLOOP
*
VNOROLL         STA   DIFF
VERTLOOP        LDA   YLOOKLO,X
                STA   HBASL
                LDA   YLOOKHI,X
                ORA   G_PAGE
                STA   HBASL+1
LLMDV           LDA   (HBASL),Y  ; PLOT PIXEL
                ORA   ANDMSK     ; SET ADDITIONAL POINTS
                STA   (HBASL),Y  ; STORE POINTS
LVCHK           CPX   #$00
                BEQ   VRTS
LVMODY          INX
                LDA   DIFF
                SEC
                SBC   DELTAXL
                BCS   VNOROLL
                ADC   DELTAY
                STA   DIFF
                LDA   ANDMSK     ; MOVE RIGHT
                ASL
                EOR   #$80
                BEQ   IS8
                STA   ANDMSK
                BNE   VERTLOOP
*
IS8             INY
                LDA   #$81
                STA   ANDMSK
                BNE   VERTLOOP
VRTS            RTS
*
WIDEDOM         LDA   DELTAXH
                STA   COUNTH
                LDX   DELTAXL
                STX   COUNT
                STX   DIFF
                LSR
                ROR   DIFF
                STA   DIFFH
                LDX   YPOSN
*
WIDELOOP        LDA   (HBASL),Y  ; PLOT PIXEL
                ORA   ANDMSK     ; SET ADDITIONAL POINTS
                STA   (HBASL),Y  ; STORE POINTS
                LDA   ANDMSK
                ASL
                EOR   #$80
```

```
                BNE   NOT7
                INY
                LDA   #$81
NOT7            STA   ANDMSK
*
                LDA   DIFF
                SEC
                SBC   DELTAY
                BCS   WNOROLL
                DEC   DIFFH
                BPL   WNOROLL
                ADC   DELTAXL
                STA   DIFF
                LDA   DIFFH
                ADC   DELTAXH
                STA   DIFFH
LWMODY          INX
                LDA   YLOOKLO,X
                STA   HBASL
                LDA   YLOOKHI,X
                ORA   G_PAGE
                STA   HBASL+1
                BNE   WDECC
*
WNOROLL         STA   DIFF
WDECC           DEC   COUNT
                LDA   COUNT
                CMP   #$FF
                BNE   WIDELOOP
                DEC   COUNTH
                BEQ   WIDELOOP
                RTS
*
* PURE VERTICAL LINE
*
VERTLINE        LDX   YDRAW
                LDY   YTO
                CPX   YTO
                BLT   USEY0
                TXA
                TAY
                LDX   YTO
USEY0           STX   YPOSN
                INY
                STY   LPVYTEST+1
                LDX   XDRAW
                LDA   XDRAW+1
                BEQ   LOTABL2
                LDY   DIV7HI,X
                LDA   MOD7HI,X
                BPL   GOTIT
LOTABL2         LDY   DIV7LO,X
                LDA   MOD7LO,X
GOTIT           TAX
                LDA   ANDMASK,X
                STA   LPVAND+1
                LDX   YPOSN
*
PVERLOOP        LDA   YLOOKLO,X
                STA   HBASL
                LDA   YLOOKHI,X
                ORA   G_PAGE
                STA   HBASL+1
LLMDPV          LDA   (HBASL),Y
```

```
LPVAND          ORA   #$00       ; SELF-MODIFYING
                STA   (HBASL),Y
                INX
LPVYTEST        CPX   #$00
                BNE   PVERLOOP
                RTS
*
                LST  OFF
*
                CHK
*
```