```
                .INSERT A:S.ASM
               @.REMARK /
               @                      *
               @                    * * *
               @                     ***
               @                   *******
               @                       *
               @                      *
               @                     *
               @                    *
               @                   *
               @                   *******
               @
               @      "WHEN YOU CARE ENOUGH TO PROGRAM
               @                THE VERY BEST"
               @
               @             ZGRASS V2.00000000
               @BY JAY FENTON, NOLA DONATO, AND TOM DEFANTI
               @                  (C) 1978
               @
               @/
                .INSERT A:ZRAM.ASM
                .INSERT A:IOEQU.ASM
                .INSERT A:NCUEQU.ASM
0001            BX.X=1
0006            BX.Y=6
000B            BX.XS=11
0010            BX.YS=16
0015            BX.MOD=21
                        .PREL
                        .IDENT  GRAPH
                        .EXTERN ALLOC
                        .EXTERN ARG
                        .EXTERN CHARWIN
                        .EXTERN DEC5IY
                        .EXTERN DIV16
                        .EXTERN DONCU
                        .EXTERN DVDE2
                        .EXTERN ERRPGM
                        .EXTERN INC5IY
                        .EXTERN IPOP
                        .EXTERN IPUSH
                        .EXTERN LN
                        .EXTERN PUTTAPE
                        .EXTERN RETNONE
                        .EXTERN SINE
                ;
                        .INTERN BOX
                        .INTERN CLEAR
                        .INTERN CPHLDE
                        .INTERN DISPLAY
                        .INTERN DOBOX
                        .INTERN LINE
                        .INTERN PIXEL
                        .INTERN R2A
```

```
                                .INTERN SCLEAR
                                .INTERN SCROLE
                                .INTERN SNAP
0012                    SNP.XS=12H
0014                    SNP.YS=14H
0016                    SNP.DS=16H
0006                    AR.TYP=6
0007                    AR.SIZ=7
0009                    AR.DIM=9
0028                    BYTEPL=40               ; BYTES PER LINE OF DISPLAY
OFFF                    URINAL=OFFFH            ; STUPID MEMORY CELL
4FFF                    CRAPPER=4FFFH
                        ; ZGRASS PIXEL FUNCTION
                        M.CMD[PIXEL,..PIXX,0,PINPUT,..PIXA,..PIXS]I
0000'   18             +PIXEL:  .BYTE $CMDADR
0001'   03             +        .BYTE (..PIXX-PIXEL)/16+1
0002'   00             +        .BYTE 0
0003'   0039'          +        .WORD PINPUT
0005'   0010'          +        .WORD ..PIXS
0007'   0013'          +        .WORD ..PIXA
0009'   504958454C00+   +        .ASCIZ /PIXEL/
                       +]
000F'   00                       .BYTE   0
0010'   CD 0000:05     ..PIXS:  CALL    ARG
0013'   040400         ..PIXA:  .BYTE   $IVAL,$IVAL,$TAF
0016'   E5                       PUSH    H
0017'   FD5606                   MOV     D,6(Y)          ; D=Y
001A'   FD5E01                   MOV     E,1(Y)          ; E=X
001D'   CD 0029'                 CALL    GPIXEL
0020'   5F                       MOV     E,A
0021'   1600                     MVI     D,0
0023'   3E04                     MVI     A,$IVAL
0025'   E1                       POP     H
0026'   C3 0000:11               JMP     RETNONE
0029'                   ..PIXX:
                        ; ROUTINE TO INTERROGATE PIXEL
0029'   CD 0682'        GPIXEL:  CALL    R2ACLP
002C'   47                       MOV     B,A
002D'   3E04                     MVI     A,4
002F'   D0                       RNC
0030'   04                       INR     B
0031'   7E                       MOV     A,M
0032'   07              ..SHFT:  RLC
0033'   07                       RLC
0034'   10FC                     DJNZ    ..SHFT
0036'   E603                     ANI     3
0038'   C9                       RET
                        M.CMD[PINPUT,..PIX,0,POUTPUT,..PIA,..PIS]I
0039'   18             +PINPUT: .BYTE $CMDADR
003A'   03             +        .BYTE (..PIX-PINPUT)/16+1
003B'   00             +        .BYTE 0
003C'   005A'          +        .WORD POUTPUT
003E'   0049'          +        .WORD ..PIS
0040'   004C'          +        .WORD ..PIA
0042'   50494E505554+   +        .ASCIZ /PINPUT/
```

```
                        +]
0049'  CD 0000:05       ..PIS:  CALL    ARG
004C'  0400             ..PIA:  .BYTE   $IVAL,$TAF
004E'  FD4E01                   MOV     C,1(Y)
0051'  ED58                     INP     E
0053'  1600                     MVI     D,0
0055'  3E04                     MVI     A,$IVAL
0057'  C3 0000:11              JMP     RETNONE
005A'                   ..PIX:
                        M.CMD[POUTPUT,..POX,0,POINT,..POA,..POS][
005A'  18               +POUTPUT:       .BYTE $CMDADR
005B'  03               +               .BYTE (..POX-POUTPUT)/16+1
005C'  00               +               .BYTE 0
005D'  007A'            +               .WORD POINT
005F'  006B'            +               .WORD ..POS
0061'  006E'            +               .WORD ..POA
0063'  504F55545055+    +               .ASCIZ /POUTPUT/
                        +]
006B'  CD 0000:05       ..POS:  CALL    ARG
006E'  040400           ..POA:  .BYTE   $IVAL,$IVAL,$TAF
0071'  FD4E01                   MOV     C,1(Y)
0074'  FD7E06                   MOV     A,6(Y)
0077'  ED79                     OUTP    A
0079'  C9                       RET
007A'                   ..POX:
                        M.CMD[POINT,..PNTX,0,PUTTAPE,..PNTA,..PNTS][
007A'  18               +POINT: .BYTE $CMDADR
007B'  03               +               .BYTE (..PNTX-POINT)/16+1
007C'  00               +               .BYTE 0
007D'  0000:10          +               .WORD PUTTAPE
007F'  008A'            +               .WORD ..PNTS
0081'  008D'            +               .WORD ..PNTA
0083'  504F494E5400+    +               .ASCIZ /POINT/
                        +]
0089'  00                       .BYTE   0
008A'  CD 0000:05       ..PNTS: CALL    ARG
008D'  04040400         ..PNTA: .BYTE   $IVAL,$IVAL,$IVAL,$TAF
0091'  FD5606                   MOV     D,6(Y)
0094'  FD5E01                   MOV     E,1(Y)
0097'  FD7E0B                   MOV     A,11(Y)
009A'  E5                       PUSH    H
009B'  CD 00A0'                 CALL    POINTR
009E'  E1                       POP     H
009F'  C9                       RET
00A0'                   ..PNTX:
                        ; POINT ROUTINE
                        ; A=MODE PARAMETER
                        ; DE= Y,X COORDINATES
                        ; HL=SMASHED
00A0'  CB57             POINTR: BIT     2,A     ; OR OR XOR?
00A2'  201F                     JRNZ    PRPLOP  ; NO
                        ; OR OR XOR - SET XPAND
00A4'  07                       RLC
00A5'  07                       RLC
00A6'  E6FC                     ANI     0FCH
```

```
00A8'  D319              OUT     XPAND
00AA'  CB6F              BIT     5,A
00AC'  2008              JRNZ    ORPT
00AE'  CD 0682'  XORPT:  CALL    R2ACLP
00B1'  D0                RNC
00B2'  F628              ORI     01010000B
00B4'  1806              JMPR    ORJOIN
00B6'  CD 0682'  ORPT:   CALL    R2ACLP
00B9'  D0                RNC
00BA'  F618              ORI     00110000B
00BC'  D30C      ORJOIN: OUT     MAGIC
00BE'  CBB4              RES     6,H
00C0'  3680              MVI     M,80H
00C2'  C9                RET
00C3'  CB5F      PRPLOP: BIT     3,A
00C5'  2029              JRNZ    PRIOR
00C7'  C5        PLOP:   PUSH    B
00C8'  D319      PLOP1:  OUT     XPAND
00CA'  3E08              MVI     A,00001000B
00CC'  D30C              OUT     MAGIC
00CE'  32 0FFF           STA     URINAL
00D1'  3A 4FFF           LDA     CRAPPER
00D4'  47                MOV     B,A
00D5'  CD 0682'          CALL    R2ACLP
00D8'  3014              JRNC    PLOPNG
00DA'  F608              ORI     00001000B
00DC'  D30C              OUT     MAGIC
00DE'  3E8C              MVI     A,10001100B
00E0'  D319              OUT     XPAND
00E2'  32 0FFF           STA     URINAL
00E5'  3A 4FFF           LDA     CRAPPER
00E8'  4F                MOV     C,A
00E9'  78                MOV     A,B
00EA'  AE                XRA     M
00EB'  A1                ANA     C
00EC'  AE                XRA     M
00ED'  77                MOV     M,A
00EE'  C1        PLOPNG: POP     B
00EF'  C9                RET
                 ; PRIORITY
00F0'  C5        PRIOR:  PUSH    B
00F1'  E603              ANI     3
00F3'  4F                MOV     C,A
00F4'  CD 0029'          CALL    GPIXEL
00F7'  B9                CMP     C       ; COMPARE TO WHATS THERE
00F8'  30F4              JRNC    PLOPNG
00FA'  79                MOV     A,C
00FB'  18CB              JMPR    PLOP1
                 ; ZGRASS CIRCLE COMMAND
                 M.CMD[CIRCLE,..CIRX,0,CHARWIN,..CIRA,..CIRS]I
00FD'  18        +CIRCLE: .BYTE $CMDADR
00FE'  0F        +        .BYTE (..CIRX-CIRCLE)/16+1
00FF'  00        +        .BYTE 0
0100'  0000:06   +        .WORD CHARWIN
0102'  010E'     +        .WORD ..CIRS
```

```
0104'   0111'        +          .WORD ..CIRA
0106'   434952434C45+            .ASCIZ /CIRCLE/
                    +]
010D'   00                       .BYTE   0
010E'   CD 0000:05    ..CIRS:    CALL    ARG
0111'   040404040400 ..CIRA:    .BYTE   $IVAL,$IVAL,$IVAL,$IVAL,$IVAL,$TAF
0001                  ..X=1
0006                  ..Y=6
000B                  ..XS=11
0010                  ..YS=16
0015                  ..MOD=21
001A                  ..BX=26
001F                  ..BY=31
0024                  ..BXS=36
0029                  ..BYS=41
002E                  ..BMOD=46
0117'   E5                       PUSH    H
0118'   FD4E10                   MOV     C,..YS(Y)
011B'   FD4611                   MOV     B,..YS+1(Y)
011E'   79                       MOV     A,C             ; GET MAD IF
011F'   B0                       ORA     B               ; YD=0
                                 ZERROR  ER.BOX[
0120'   2004          +          JRNZ ..0001
0122'   CD 0000:0B    +          CALL ERRPGM
0125'   1A            +          .BYTE ER.BOX
0126'                 +..0001:]
0126'   FD5E0B                   MOV     E,..XS(Y)       ; SIMILARLY
0129'   FD560C                   MOV     D,..XS+1(Y)     ; IF XD=0
012C'   7B                       MOV     A,E
012D'   B2                       ORA     D
                                 ZERROR  ER.BOX[
012E'   2004          +          JRNZ ..0002
0130'   CD 0000:0B    +          CALL ERRPGM
0133'   1A            +          .BYTE ER.BOX
0134'                 +..0002:]
0134'   59                       MOV     E,C
0135'   50                       MOV     D,B
0136'   CD 0000:0A               CALL    DVDE2   ; DE=DE/2
0139'   C5             ..CLP:    PUSH    B
013A'   D5                       PUSH    D
                       ; START CRUNCHIN...
013B'   3E01                     MVI     A,1             ; PUSH 1.0
013D'   CD 01E9'                 CALL    PUTSCN
                       ;
0140'   CD 0000:0E               CALL    IPUSH           ; PUT Y ON STACK
0143'   3E1D                     MVI     A,N.FLTS        ; FLOAT IT
0145'   CD 0000:09               CALL    DONCU
                       ;
0148'   FD5E10                   MOV     E,..YS(Y)       ; PUT YD ON STACK
014B'   FD5611                   MOV     D,..YS+1(Y)
014E'   CD 0000:0E               CALL    IPUSH
0151'   3E1D                     MVI     A,N.FLTS        ; FLOAT YD TOO
0153'   CD 0000:09               CALL    DONCU
                       ;
0156'   3E13                     MVI     A,N.FDIV        ; DO DIVIDE
```

```
0158'  CD 0000:09          CALL    DONCU
                ;
015B'  3E02                MVI     A,2             ; PUSH 2.0
015D'  CD 01E9'            CALL    PUTSCN
                ;
0160'  3E12                MVI     A,N.FMUL        ; MULTIPLY
0162'  CD 0000:09          CALL    DONCU
                ;
0165'  3E17                MVI     A,N.PTOF        ; DUPLICATE TOP
0167'  CD 0000:09          CALL    DONCU
                ;
016A'  3E12                MVI     A,N.FMUL        ; MULTIPLY TO SQUARE
016C'  CD 0000:09          CALL    DONCU
                ;
016F'  3E11                MVI     A,N.FSUB
0171'  CD 0000:09          CALL    DONCU
                ;
0174'  3E01                MVI     A,N.SQRT        ; DO SQRT
0176'  CD 0000:09          CALL    DONCU
                ;
0179'  FD5E0B              MOV     E,..XS(Y)       ; PUSH XD
017C'  FD560C              MOV     D,..XS+1(Y)
017F'  CD 0000:0E          CALL    IPUSH
0182'  3E1D                MVI     A,N.FLTS
0184'  CD 0000:09          CALL    DONCU
                ;
0187'  3E12                MVI     A,N.FMUL
0189'  CD 0000:09          CALL    DONCU
                ;
018C'  AF                  XRA     A               ; PUSH .5
018D'  CD 01E9'            CALL    PUTSCN
                ;
0190'  3E10                MVI     A,N.FADD
0192'  CD 0000:09          CALL    DONCU
                ;
0195'  3E1F                MVI     A,N.FIXS
0197'  CD 0000:09          CALL    DONCU
                ;
019A'  CD 0000:0D          CALL    IPOP            ; WE HAVE DE=XS
                ; IF XS = 0, SKIP IT
019D'  7A                  MOV     A,D
019E'  B3                  ORA     E
019F'  283D                JRZ     ..SKIP
01A1'  CBC3                SET     0,E             ; FORCE TO BE ODD!
                ; BUILD CALL TO BOX
01A3'  FD7324              MOV     ..BXS(Y),E      ; STUFF XS
01A6'  FD7225              MOV     ..BXS+1(Y),D
                ; TRANSFER REMAINING CRAP
01A9'  FD5E01              MOV     E,..X(Y)
01AC'  FD5602              MOV     D,..X+1(Y)      ; LIKE X
01AF'  FD731A              MOV     ..BX(Y),E
01B2'  FD721B              MOV     ..BX+1(Y),D
                ;
01B5'  FD362901            MVI     ..BYS(Y),1      ; SET YS
01B9'  FD362A00            MVI     ..BYS+1(Y),0
```

```
                        ;
01BD'   FD7E15                  MOV     A,..MOD(Y)
01C0'   FD772E                  MOV     ..BMOD(Y),A
                        ;
01C3'   FD6E06                  MOV     L,..Y(Y)
01C6'   FD6607                  MOV     H,..Y+1(Y)
01C9'   D1                      POP     D
01CA'   D5                      PUSH    D
01CB'   19                      DAD     D
01CC'   FD751F                  MOV     ..BY(Y),L
01CF'   FD7420                  MOV     ..BY+1(Y),H
01D2'   FDE5                    PUSH    Y
01D4'   11 0019                 LXI     D,25
01D7'   FD19                    DADY    D
01D9'   CD 0434'                CALL    DOBOX
01DC'   FDE1                    POP     Y
01DE'   D1              ..SKIP: POP     D
01DF'   1B                      DCX     D
01E0'   C1                      POP     B
01E1'   0B                      DCX     B
01E2'   78                      MOV     A,B
01E3'   B1                      ORA     C
01E4'   C2 0139'                JNZ     ..CLP
01E7'   E1                      POP     H
01E8'   C9                      RET
01E9'                   ..CIRX:
                        ; SUBROUTINE TO PUT SMALL CONSTANT ON NCU STACK
01E9'   D5              PUTSCN: PUSH    D
01EA'   F5                      PUSH    PSW
01EB'   11 0000                 LXI     D,0
01EE'   CD 0000:0E              CALL    IPUSH
01F1'   1E80                    MVI     E,80H
01F3'   F1                      POP     PSW
01F4'   57                      MOV     D,A
01F5'   CD 0000:0E              CALL    IPUSH
01F8'   D1                      POP     D
01F9'   C9                      RET
                        ; ZGRASS SCROLL COMMAND
                        M.CMD[SCROLL,SCROLX,0,SINE,SCROLA,SCROLS][
01FA'   18              +SCROLL: .BYTE $CMDADR
01FB'   09              +        .BYTE (SCROLX-SCROLL)/16+1
01FC'   00              +        .BYTE 0
01FD'   0000:12         +        .WORD SINE
01FF'   020B'           +        .WORD SCROLS
0201'   020E'           +        .WORD SCROLA
0203'   5343524F4C4C+            .ASCIZ /SCROLL/
                        +]
020A'   00                      .BYTE   0
020B'   CD 0000:05      SCROLS: CALL    ARG
020E'   040404040400    SCROLA: .BYTE   $IVAL,$IVAL,$IVAL,$IVAL,$IVAL,$TAF
0214'   E5              SCROLE: PUSH    H
0215'   11 0050                 LXI     D,80
0218'   21 FFB1                 LXI     H,-79
021B'   CD 04F0'                CALL    CLIPPER
021E'   3860                   JRC     ..NOSC
```

```
0220'  FDE5                PUSH    Y
0222'  CD 0000:0C          CALL    INC5IY
0225'  11 0033             LXI     D,51
0228'  21 FFCE             LXI     H,-50
022B'  CD 04F0'            CALL    CLIPPER
022E'  FDE1                POP     Y
0230'  384E                JRC     ..NOSC
                     ; CONVERT X SIZE TO BYTES
0232'  FD7E0B              MOV     A,BX.XS(Y)
0235'  C603                ADI     3
0237'  0F                  RRC
0238'  0F                  RRC
0239'  E63F                ANI     3FH
023B'  4F                  MOV     C,A
                     ; WHICH DIRECTION?
023C'  FD7E15              MOV     A,BX.MOD(Y)
023F'  A7                  ANA     A
0240'  283E                JRZ     ..NOSC
0242'  11 FFD8             LXI     D,-40
0245'  FD4610              MOV     B,BX.YS(Y)
0248'  FD7E06              MOV     A,BX.Y(Y)
024B'  FA 0253'            JM      ..MINU
                     ; POSITIVE CASE
024E'  11 0028             LXI     D,40
0251'  80                  ADD     B
0252'  3D                  DCR     A
0253'  05            ..MINU: DCR   B         ; FUDGE Y SIZE
0254'  282A                JRZ     ..NOSC  ; SKIP IF WAS ONLY 1
0256'  D5                  PUSH    D
0257'  57                  MOV     D,A
0258'  FD5E01              MOV     E,BX.X(Y)
025B'  CD 0690'            CALL    R2A
025E'  D1                  POP     D
025F'  FD7E15              MOV     A,BX.MOD(Y)
0262'  CD 067D'            CALL    ABS
0265'  C5            ..SCR1: PUSH  B
0266'  E5                  PUSH    H
0267'  C5           ..SCR2: PUSH   B
0268'  D5                  PUSH    D
0269'  0600                MVI     B,0
026B'  EB                  XCHG
026C'  19                  DAD     D
026D'  E5                  PUSH    H
026E'  EDB0                LDIR
0270'  E1                  POP     H
0271'  D1                  POP     D
0272'  C1                  POP     B
0273'  10F2                DJNZ    ..SCR2
0275'  3600          ..KILL: MVI   M,0
0277'  23                  INX     H
0278'  0D                  DCR     C
0279'  20FA                JRNZ    ..KILL
027B'  E1                  POP     H
027C'  C1                  POP     B
027D'  3D                  DCR     A
```

```
027E'   20E5                    JRNZ    ..SCR1
0280'   E1              ..NOSC: POP     H
0281'   C9                      RET
0282'                   SCROLX:
                        ; ZGRASS SNAP COMMAND
                        M.CMD[SNAP,..SNPX,0,SCROLL,..SNPA,..SNPS][
0282'   18              +SNAP:  .BYTE $CMDADR
0283'   12              +       .BYTE (..SNPX-SNAP)/16+1
0284'   00              +       .BYTE 0
0285'   01FA'           +       .WORD SCROLL
0287'   0291'           +       .WORD ..SNPS
0289'   0294'           +       .WORD ..SNPA
028B'   534E415000      +       .ASCIZ /SNAP/
                        +]
0290'   00                      .BYTE   0
0025                    ..OVH=SNP.DS+0FH        ; OVERHEAD FOR ALLOCATION
0001                    ..NAME=1                ; IY NAME
0006                    ..X=6
000B                    ..Y=11
0010                    ..XS=16
0015                    ..YS=21
0291'   CD 0000:05      ..SNPS: CALL    ARG
0294'   0A0404040400    ..SNPA: .BYTE   $NAME,$IVAL,$IVAL,$IVAL,$IVAL,$TAF
029A'   E5                      PUSH    H
029B'   DDE5                    PUSH    X
029D'   FDE5                    PUSH    Y
                        ; CLIP THE BOX
029F'   CD 0000:0C              CALL    INC5IY
02A2'   11 0050                 LXI     D,80
02A5'   21 FFB1                 LXI     H,-79
02A8'   CD 04F0'                CALL    CLIPPER
                                CERROR  ER.SNP[
02AB'   3004            +       JRNC ..0003
02AD'   CD 0000:0B      +       CALL ERRPGM
02B0'   31              +       .BYTE ER.SNP
02B1'                   +..0003:]
02B1'   CD 0000:0C              CALL    INC5IY
02B4'   11 0033                 LXI     D,51
02B7'   21 FFCE                 LXI     H,-50
02BA'   CD 04F0'                CALL    CLIPPER
                                CERROR  ER.SNP[
02BD'   3004            +       JRNC ..0004
02BF'   CD 0000:0B      +       CALL ERRPGM
02C2'   31              +       .BYTE ER.SNP
02C3'                   +..0004:]
02C3'   FDE1                    POP     Y       ; BACK TO NORMAL
                        ; FIGURE OUT HOW MUCH MEMORY WE NEED
02C5'   FD7E10                  MOV     A,..XS(Y)       ; GET X SIZE
02C8'   C603                    ADI     3               ; + 3
02CA'   0F                      RRC
02CB'   0F                      RRC
02CC'   E63F                    ANI     3FH
02CE'   6F                      MOV     L,A             ; TO L
02CF'   2600                    MVI     H,0
02D1'   FD5E15                  MOV     E,..YS(Y)
```

```
02D4'  54              MOV     D,H
02D5'  CD 039D'        CALL    JMUL
02D8'  E5              PUSH    H
02D9'  11 0025         LXI     D,..OVH
02DC'  19              DAD     D
02DD'  CD 0000:08      CALL    DIV16
02E0'  7D              MOV     A,L
02E1'  CD 0000:04      CALL    ALLOC
02E4'  E5              PUSH    H               ; IX=ARRAY HEADER ADDR
02E5'  DDE1            POP     X
02E7'  D1              POP     D               ; DE=BYTES REQUESTED
02E8'  E5              PUSH    H
02E9'  AF              XRA     A
02EA'  DD7702          MOV     $FLAGS(X),A     ; CLEAR FLAGS
02ED'  DD360010        MVI     $TYPE(X),$AADR  ; LABEL AS ARRAY
02F1'  DD360501        MVI     $USE(X),1       ; USE COUNT OF 1
02F5'  DD360605        MVI     AR.TYP(X),$ADDRI        ; TYPE INTEGER
02F9'  21 0004         LXI     H,4
02FC'  19              DAD     D
02FD'  DD7507          MOV     AR.SIZ(X),L     ; SET SIZE
0300'  DD7408          MOV     AR.SIZ+1(X),H
0303'  DD7509          MOV     AR.DIM(X),L     ; AND DIMENSION
0306'  DD740A          MOV     AR.DIM+1(X),H
0309'  DD360BFF        MVI     AR.DIM+2(X),0FFH        ; SET TERMINATOR

030D'  FD5615          MOV     D,..YS(Y)       ; SET FIRST ELEMENTS
0310'  FD5E10          MOV     E,..XS(Y)
0313'  DD7312          MOV     SNP.XS(X),E     ; TO X, Y SIZES IN PIXEL
                       S
0316'  DD7713          MOV     SNP.XS+1(X),A
0319'  DD7214          MOV     SNP.YS(X),D
031C'  DD7715          MOV     SNP.YS+1(X),A
031F'  FDE5            PUSH    Y
0321'  7B              MOV     A,E
0322'  C603            ADI     3
0324'  0F              RRC
0325'  0F              RRC
0326'  E63F            ANI     3FH
0328'  5F              MOV     E,A
0329'  D5              PUSH    D
032A'  FD5E06          MOV     E,..X(Y)        ; E=X
032D'  FD7E0B          MOV     A,..Y(Y)        ; FUDGE Y TO TOPMOST LIN
                       E
0330'  82              ADD     D
0331'  3D              DCR     A
0332'  57              MOV     D,A
0333'  01 0016         LXI     B,SNP.DS        ; IX=DEST AREA ADDR
0336'  DD09            DADX    B
0338'  CD 0690'        CALL    R2A
033B'  E5              PUSH    H               ; IY=SOURCE POINTER
033C'  FDE1            POP     Y
033E'  4F              MOV     C,A             ; C=SHIFT AMOUNT
033F'  D1              POP     D               ; DE=SIZES
0340'  2827            JRZ     ..EASY          ; JUMP ON EASY CASE
                ; CASE OF FIRST PIXEL TO SNAP NOT BEING ON BYTE BOUNDARY
```

```
0342'   D5              ..HARD: PUSH    D
0343'   FDE5                   PUSH    Y
0345'   FD6600                 MOV     H,0(Y)          ; H=FIRST BYTE TO START
0348'   FD23            ..HDBL: INX     Y               ; BUMP SOURCE
034A'   FD5600                  MOV     D,0(Y)          ; D=NEXT GUY
034D'   6A                      MOV     L,D             ; INTO L AS WELL
034E'   41                      MOV     B,C             ; REINIT SHIFT AMT
034F'   29              ..HDSL: DAD     H               ; SHIFT OVER SA PIXELS
0350'   29                      DAD     H
0351'   10FC                    DJNZ    ..HDSL
0353'   DD7400                  MOV     0(X),H          ; STUFF TO DEST
0356'   62                      MOV     H,D             ; SETUP FOR NEXT ITERATI
                        ON
0357'   DD23                    INX     X
0359'   1D                      DCR     E
035A'   20EC                    JRNZ    ..HDBL
035C'   FDE1                    POP     Y               ; TO NEXT LINE
035E'   11 0028                 LXI     D,40
0361'   FD19                    DADY    D
0363'   D1                      POP     D
0364'   15                      DCR     D
0365'   20DB                    JRNZ    ..HARD
0367'   181A                    JMPR    ..SNPD
                        ; FASTER LOOP FOR ZERO SHIFT AMOUNT CASE
0369'   01 0028          ..EASY: LXI     B,40
036C'   7B              ..EZLL: MOV     A,E
036D'   FDE5                    PUSH    Y
036F'   FD6E00           ..EZBL: MOV     L,0(Y)
0372'   DD7500                  MOV     0(X),L
0375'   DD23                    INX     X
0377'   FD23                    INX     Y
0379'   3D                      DCR     A
037A'   20F3                    JRNZ    ..EZBL
037C'   FDE1                    POP     Y
037E'   FD09                    DADY    B
0380'   15                      DCR     D
0381'   20E9                    JRNZ    ..EZLL
                        ;
                        ; DONE WITH TRANSFER - MAKE NAME POINT AT ME
0383'   FDE1            ..SNPD: POP     Y       ; IY BACK TO NORMAL
0385'   FD6602                  MOV     H,..NAME+1(Y)   ; GET PTR TO NAME IN IX
0388'   FD6E01                  MOV     L,..NAME(Y)
038B'   E5                      PUSH    H
038C'   DDE1                    POP     X
038E'   D1                      POP     D
038F'   DD36000A                MVI     $TYPE(X),$NAMADR
0393'   DD7305                  MOV     $NVALUE(X),E
0396'   DD7206                  MOV     $NVALUE+1(X),D
0399'   DDE1                    POP     X
039B'   E1                      POP     H
039C'   C9                      RET
039D'                   ..SNPX:
                        ; MULTIPLY HL BY DE ROUTINE
                        ; SLOW AND SLEAZY
```

```
039D'   C5            JMUL:   PUSH    B
039E'   7C                    MOV     A,H
039F'   A7                    ANA     A
03A0'   2801                  JRZ     ..NOXC
03A2'   EB                    XCHG
03A3'   45            ..NOXC: MOV     B,L
03A4'   21 0000               LXI     H,0
03A7'   19            ..MPYL: DAD     D
03A8'   10FD                  DJNZ    ..MPYL
03AA'   C1                    POP     B
03AB'   C9                    RET
                      ; ZGRASS DISPLAY COMMAND
                      M.CMD[DISPLAY,..DISX,0,AFIX[D],..DISA,..DISS][
03AC'   18           +DISPLAY:        .BYTE $CMDADR
03AD'   08           +        .BYTE (..DISX-DISPLAY)/16+1
03AE'   00           +        .BYTE 0
03AF'   631C         +        .WORD AFIX[D][("D"-101Q)*16+FSTINT]
03B1'   03BE'        +        .WORD ..DISS
03B3'   03C1'        +        .WORD ..DISA
03B5'   444953504C41+         .ASCIZ /DISPLAY/
                     +]
03BD'   00                    .BYTE   0
03BE'   CD 0000:05    ..DISS: CALL    ARG
03C1'   1004040400    ..DISA: .BYTE   $AADR,$IVAL,$IVAL,$IVAL,$TAF
0001                  ..NAM=1
0006                  ..X=6
000B                  ..Y=11
0010                  ..MODE=16
03C6'   E5                    PUSH    H
03C7'   DDE5                  PUSH    X
                      ; IX=ARRAY HEADER ADDR
03C9'   FD6E01                MOV     L,..NAM(Y)
03CC'   FD6602                MOV     H,..NAM+1(Y)
03CF'   E5                    PUSH    H
03D0'   DDE1                  POP     X
03D2'   01 0016               LXI     B,SNP.DS
03D5'   09                    DAD     B
03D6'   E5                    PUSH    H               ; SAVE FOR POSTERITY
03D7'   DD7E12                MOV     A,SNP.XS(X)     ; GET XS
03DA'   4F                    MOV     C,A
03DB'   3D                    DCR     A
03DC'   0F                    RRC                     ; DIVIDE BY 2
03DD'   E67F                  ANI     7FH
03DF'   5F                    MOV     E,A             ; FOR FUDGE FACTOR
03E0'   79                    MOV     A,C
03E1'   C603                  ADI     3
03E3'   0F                    RRC
03E4'   0F                    RRC                     ; DIVIDE BY 4 BY NOW
03E5'   E63F                  ANI     3FH
03E7'   4F                    MOV     C,A             ; FOR X SIZE
03E8'   FD7E06                MOV     A,..X(Y)        ; CONFUSE THE X COORDIN
                      TE
03EB'   93                    SUB     E
03EC'   5F                    MOV     E,A
03ED'   DD4614                MOV     B,SNP.YS(X)     ; SIMILAR TRICKS FOR Y
```

```
03F0'   78                      MOV     A,B
03F1'   0F                      RRC
03F2'   E67F                    ANI     7FH
03F4'   FD860B                  ADD     ..Y(Y)
03F7'   57                      MOV     D,A
03F8'   CD 0690'                CALL    R2A
03FB'   57                      MOV     D,A             ; D=SHIFT AMOUNT
03FC'   FD7E10                  MOV     A,..MODE(Y)
03FF'   07                      RLC
0400'   07                      RLC
0401'   E67C                    ANI     7CH
0403'   B2                      ORA     D
0404'   D30C                    OUT     MAGIC
0406'   CBB4                    RES     6,H
0408'   D1                      POP     D
0409'   AF                      XRA     A
                        ; NORMAL? WRITE
040A'   C5              ..MWRT: PUSH    B
040B'   EB                      XCHG
040C'   D5                      PUSH    D
040D'   47                      MOV     B,A
040E'   EDB0                    LDIR
0410'   12                      STAX    D
0411'   D1                      POP     D
0412'   EB                      XCHG
0413'   0E28                    MVI     C,BYTEPL
0415'   09                      DAD     B
0416'   C1                      POP     B
0417'   10F1                    DJNZ    ..MWRT
0419'   DDE1                    POP     X
041B'   E1                      POP     H
041C'   C9                      RET
041D'                   ..DISX:
                        ; ZGRASS BOX COMMAND
                        M.CMD[BOX,BOXX,0,AFIX[B],BOXA,BOXS][
041D'   18              +BOX:   .BYTE $CMDADR
041E'   0E              +       .BYTE (BOXX-BOX)/16+1
041F'   00              +       .BYTE 0
0420'   62FC            +       .WORD AFIX[B][("B"-101Q)*16+FSTINT]
0422'   042B'            +       .WORD BOXS
0424'   042E'            +       .WORD BOXA
0426'   424F5800        +       .ASCIZ /BOX/
                        +]
042A'   00                      .BYTE   0
042B'   CD 0000:05      BOXS:   CALL    ARG
042E'   040404040400    BOXA:   .BYTE   $IVAL,$IVAL,$IVAL,$IVAL,$IVAL,$TAF
0434'   E5              DOBOX:  PUSH    H
0435'   11 0050                 LXI     D,80
0438'   21 FFB1                 LXI     H,-79
043B'   CD 04F0'                CALL    CLIPPER         ; CLIP X
043E'   3810                    JRC     ..SKPL          ; ABORT IF TOTAL OFFSCRE
                        EN
0440'   FDE5                    PUSH    Y
0442'   CD 0000:0C              CAL@    INC5IY
0445'   11 0033                 LXI     D,51
```

```
0448'  21 FFCE            LXI     H,-50
044B'  CD 04F0'           CALL    CLIPPER         ; CLIP Y
044E'  FDE1               POP     Y
0450'  386F          ..SKPL: JRC     ..SKIP          ; SIMILARLY ABORT Y
                     ; WE NOW HAVE REASONABLE STUFF ON OUR STACK
                     ; LETS DEAL WITH MODE STUFF NOW
0452'  FD4E15             MOV     C,BX.MOD(Y)
0455'  79                 MOV     A,C             ; IS MODE ZERO?
0456'  A7                 ANA     A
0457'  2868               JRZ     ..SKIP          ; YEP - IGNORE
0459'  E604               ANI     4               ; ISOLATE WRITE MODE
045B'  32 65EE            STA     WRMODE
045E'  79                 MOV     A,C             ; ISOLATE PIXEL NUMBER
045F'  E603               ANI     3
0461'  4F                 MOV     C,A
0462'  0600               MVI     B,0             ; LOOKUP BYTE OF THOSE G
UYS
0464'  21 05FE'           LXI     H,MSKTBL
0467'  09                 DAD     B
0468'  7E                 MOV     A,M
0469'  32 65ED            STA     PIXVAL
                     ; NOW THE EXCITING BOX PAINTING STARTS
046C'  FD5E0B             MOV     E,BX.XS(Y)
                     ; WHILE XS > 0 DO [
046F'  7B            ..BOXP: MOV     A,E
0470'  A7                 ANA     A
0471'  284E               JRZ     ..SKIP
                     ; IS MOD(X,4)=1?
0473'  FD7E01             MOV     A,BX.X(Y)
0476'  E603               ANI     3
0478'  FE01               CPI     1
047A'  200E               JRNZ    ..MNZ
                     ; YES - IS XS>4?
047C'  7B                 MOV     A,E
047D'  FE04               CPI     4
047F'  3834               JRC     ..XSL4
                     ; YS IS >4 - SO PAINT A FULL STRIPE
0481'  0EFF               MVI     C,0FFH          ; DO WHOLE KIT AND KABOO
DLE
0483'  CD 04C3'           CALL    ..STRC
0486'  1604               MVI     D,4             ; A=X ADDR SUBTRACTOR
0488'  181F               JMPR    ..XSTF
048A'  3D            ..MNZ:  DCR     A
048B'  E603               ANI     3
048D'  4F                 MOV     C,A
048E'  3E04               MVI     A,4             ; COMPUTE MIN[X,4-MOD(XS
,4)]
0490'  91                 SUB     C
0491'  BB                 CMP     E               ; COMPARE TO XS
0492'  3801               JRC     ..XSBG
0494'  7B                 MOV     A,E             ; MOD IS BIGGER
0495'  47            ..XSBG: MOV     B,A             ; B=MIN
0496'  57                 MOV     D,A
0497'  AF                 XRA     A               ; FORM BIT MASK
0498'  0F            ..BITF: RRC
```

```
0499'   0F                      RRC
049A'   F6C0                    ORI     11000000B
049C'   10FA                    DJNZ    ..BITF
049E'   41                      MOV     B,C
049F'   0F              ..DOSF: RRC                     ; SHIFT OVER MOD[X,4] TI
                        MES
04A0'   0F                      RRC
04A1'   E63F                    ANI     3FH
04A3'   10FA                    DJNZ    ..DOSF
04A5'   4F                      MOV     C,A             ; REMEMBERIZE
04A6'   CD 04C3'                CALL    ..STRC
04A9'   7A              ..XSTF: MOV     A,D
04AA'   FD8601                  ADD     BX.X(Y)         ; UPDATE X COORDINATE
04AD'   FD7701                  MOV     BX.X(Y),A
04B0'   7B                      MOV     A,E             ; AND PIXELS LEFT (XS)
04B1'   92                      SUB     D
04B2'   5F                      MOV     E,A
04B3'   18BA                    JMPR    ..BOXP          ; LOOP BACK FOR MORE
                        ; PAINT A FINAL STRIPE
04B5'   47              ..XSL4: MOV     B,A
04B6'   AF                      XRA     A
04B7'   0F              ..XSLA: RRC
04B8'   0F                      RRC
04B9'   F6C0                    ORI     11000000B
04BB'   10FA                    DJNZ    ..XSLA
04BD'   4F                      MOV     C,A
04BE'   CD 04C3'                CALL    ..STRC
04C1'   E1              ..SKIP: POP     H
04C2'   C9                      RET
                        ; LOOP TO PAINT A BOX STRIPE
                        ; MASK TO USE PASSED IN C
04C3'   D5              ..STRC: PUSH    D
04C4'   FD5606                  MOV     D,BX.Y(Y)
04C7'   FD5E01                  MOV     E,BX.X(Y)
04CA'   CD 0690'                CALL    R2A             ; CONVERT COORDINATES
04CD'   11 FFD8                 LXI     D,-40           ; NEGATIVE SCREEN INCREM
                        ENT
04D0'   FD4610                  MOV     B,BX.YS(Y)      ; B=Y SIZE
04D3'   3A 65EE                 LDA     WRMODE          ; WHICH TIGHT LOOP TO US
                        E?
04D6'   A7                      ANA     A
04D7'   200B                    JRNZ    ..PLOP
                        ; WRITE MAGIC (XOR FOR NOW)
04D9'   3A 65ED         ..XORL: LDA     PIXVAL
04DC'   A1                      ANA     C
04DD'   AE                      XRA     M
04DE'   77                      MOV     M,A
04DF'   19                      DAD     D
04E0'   10F7                    DJNZ    ..XORL
04E2'   D1                      POP     D
04E3'   C9                      RET
                        ; PLOP WRITE LOOP
04E4'   3A 65ED         ..PLOP: LDA     PIXVAL
04E7'   AE                      XRA     M
04E8'   A1                      ANA     C
```

```
04E9'  AE                      XRA     M
04EA'  77                      MOV     M,A
04EB'  19                      DAD     D
04EC'  10F6                    DJNZ    ..PLOP
04EE'  D1                      POP     D
04EF'  C9                      RET
04F0'                  BOXX:
                       ; CLIP COORDINATE ROUTINE
                       ; HL=LOWER LIMIT
                       ; DE=UPPER LIMIT
                       ; IY=POINTS TO TYPE OF COORDINATE ON IY STACK
04F0'  E5              CLIPPER:        PUSH    H
04F1'  FD6E01                  MOV     L,1(Y)          ; HL=COORDINATE
04F4'  FD6602                  MOV     H,2(Y)
04F7'  FD4E0B                  MOV     C,11(Y)         ; BC=SIZE
04FA'  FD460C                  MOV     B,12(Y)
04FD'  CD 0580'                CALL    ..TSTB          ; BARF IF <= 0
                               ZERROR  ER.BOX[
0500'  2004            +       JRNZ ..0005
0502'  CD 0000:0B      +       CALL ERRPGM
0505'  1A              +       .BYTE ER.BOX
0506'                  +..0005:]
0506'  0B                     DCX     B
0507'  CD 0590'                CALL    ..DVBC          ; BC=BC DIVIDE 2
050A'  A7                      ANA     A
050B'  ED42                    DSBC    B               ; HL=LOWER
050D'  FD7501                  MOV     1(Y),L          ; STUFF BACK STUFF
0510'  FD7402                  MOV     2(Y),H
0513'  EB                      XCHG                    ; TO DE
0514'  CD 0598'                CALL    CPHLDE          ; IS LOWER>UL?
0517'  3864                    JRC     ..NODR          ; DONT DRAW
0519'  E3                      XTHL
051A'  CD 0598'                CALL    CPHLDE          ; IS LOWER < LOWER LIMIT
                       ?
051D'  381F                    JRC     ..LOK
051F'  E5                      PUSH    H               ; SAVE LOWER LIMIT
0520'  EB                      XCHG                    ; HL=LOWER,DE=LL
0521'  A7                      ANA     A
0522'  ED52                    DSBC    D               ; HL=LOWER - LIMIT
0524'  EB                      XCHG
0525'  FD6E0B                  MOV     L,11(Y)         ; HL=SIZE
0528'  FD660C                  MOV     H,12(Y)
052B'  19                      DAD     D
052C'  FD750B                  MOV     11(Y),L         ; STORE BACK
052F'  FD740C                  MOV     12(Y),H
0532'  CD 0589'                CALL    ..TSTH          ; IF H<= 0 ABORT
0535'  2845                    JRZ     ..NOD1
0537'  E1                      POP     H               ; SET COORDINATE AT
0538'  FD7501                  MOV     1(Y),L          ; LOWER LIMIT
053B'  FD7402                  MOV     2(Y),H
                       ; DEAL WITH OTHER END
053E'  EB              ..LOK:  XCHG            ; DE=LOWER LIMIT
053F'  FD6E01                  MOV     L,1(Y)          ; HL=COORDINATE
0542'  FD6602                  MOV     H,2(Y)
0545'  FD4E0B                  MOV     C,11(Y)         ; BC=SIZE
```

```
0548'   FD460C              MOV     B,12(Y)
054B'   CD 0580'             CALL    ..TSTB
                             ZERROR  ER.BOX[
054E'   2004          +      JRNZ ..0006
0550'   CD 0000:0B    +      CALL ERRPGM
0553'   1A            +      .BYTE ER.BOX
0554'                 +..0006:]
0554'   0B                   DCX     B
0555'   09                   DAD     B               ; ADD TO LOWER EDGE
0556'   EB                   XCHG                    ; UPPER TO DE
0557'   CD 0598'             CALL    CPHLDE          ; CAN WE DRAW?
055A'   E1                   POP     H               ; H=UL
055B'   281D                 JRZ     ..UOK           ; JUMP IF ON EDGE
055D'   301F                 JRNC    ..NOD2          ; IF UPPER < LOWER LIMIT
                      DONT
055F'   CD 0598'             CALL    CPHLDE          ; IS UPPER > UL?
0562'   3016                 JRNC    ..UOK           ; NO PROB
0564'   A7                   ANA     A               ; COMPUTE SIZE FUDGE
0565'   ED52                 DSBC    D
0567'   EB                   XCHG                    ; TO DE
0568'   FD6E0B               MOV     L,11(Y)
056B'   FD660C               MOV     H,12(Y)
056E'   19                   DAD     D               ; HL=NEW SIZE
056F'   FD750B               MOV     11(Y),L
0572'   FD740C               MOV     12(Y),H
0575'   CD 0589'             CALL    ..TSTH  ; IF HL<=0 ABORT
0578'   2804                 JRZ     ..NOD2
057A'   A7            ..UOK: ANA     A               ; RETURN CARRY CLEAR
057B'   C9                   RET                     ; FOR GOOD GUYS
057C'   E1            ..NOD1: POP     H               ; BAD GUY - CLEAN UP STA
                      CK
057D'   E1            ..NODR: POP     H
057E'   37            ..NOD2: STC                     ; CY FOR DONT DRAW
057F'   C9                   RET
                      ; TEST FOR BC BEING <= 0
0580'   78            ..TSTB: MOV     A,B
0581'   A7                   ANA     A
0582'   FA 0587'             JM      ..LESZ
0585'   B1                   ORA     C
0586'   C9                   RET
0587'   AF            ..LESZ: XRA     A
0588'   C9                   RET
                      ; SIMILAR ROUTINE FOR HL
0589'   7C            ..TSTH: MOV     A,H
058A'   A7                   ANA     A
058B'   FA 0587'             JM      ..LESZ
058E'   B5                   ORA     L
058F'   C9                   RET
                      ; DIVIDE BC BY 2
0590'   A7            ..DVBC: ANA     A
0591'   78                   MOV     A,B
0592'   1F                   RAR
0593'   47                   MOV     B,A
0594'   79                   MOV     A,C
0595'   1F                   RAR
```

```
0596'  4F                      MOV     C,A
0597'  C9                      RET
                        ; ROUTINE TO COMPARE HL TO DE
                        ; RETURNS CY SET FOR HL<DE (OR DE>HL)
                        ;         CY CLEAR, ZERO SET IF HL=DE
                        ;         CY CLEAR, ZERO CLEAR IF HL>DE (OR DE<HL)
0598'  7C               CPHLDE: MOV     A,H
0599'  AA                       XRA     D               ; ARE SIGNS DIFF?
059A'  F2 05A3'                 JP      ..CK1           ; NO
059D'  EB                       XCHG                    ; YES - REVERSE ARGS
059E'  CD 05A3'                 CALL    ..CK1           ; DO CHECK
05A1'  EB                       XCHG                    ; BACK TO NORMAL
05A2'  C9                       RET
05A3'  7C               ..CK1:  MOV     A,H
05A4'  BA                       CMP     D
05A5'  C0                       RNZ
05A6'  7D                       MOV     A,L
05A7'  BB                       CMP     E
05A8'  C9                       RET
                        ; ZGRASS CLEAR SCREEN COMMAND
                                M.CMD[CLEAR,GRCLRX,0,CIRCLE,0,SCLEAR]I
05A9'  18              +CLEAR:  .BYTE $CMDADR
05AA'  03              +        .BYTE (GRCLRX-CLEAR)/16+1
05AB'  00              +        .BYTE 0
05AC'  00FD'           +        .WORD CIRCLE
05AE'  05B9'           +        .WORD SCLEAR
05B0'  0000            +        .WORD 0
05B2'  434C45415200+           .ASCIZ /CLEAR/
                       +]
05B8'  00                      .BYTE   0
05B9'  E5               SCLEAR: PUSH    H
05BA'  21 4000                  LXI     H,4000H
05BD'  75                       MOV     M,L
05BE'  11 4001                  LXI     D,4001H
05C1'  01 0FFF                  LXI     B,0FFFH
05C4'  EDB0                     LDIR
05C6'  ED43 65EF                SBCD    OLDXY   ; RESET LINE PTR
05CA'  E1                       POP     H
05CB'  C9                       RET
05CC'                   GRCLRX:
                        ; ZGRASS LINE COMMAND
                                M.CMD   [LINE,..LINX,0,LN,..LINA,..LIN1]I
05CC'  18              +LINE:   .BYTE $CMDADR
05CD'  04              +        .BYTE (..LINX-LINE)/16+1
05CE'  00              +        .BYTE 0
05CF'  0000:0F         +        .WORD LN
05D1'  05DB'           +        .WORD ..LIN1
05D3'  05DE'           +        .WORD ..LINA
05D5'  4C494E4500      +        .ASCIZ /LINE/
                       +]
05DA'  00                      .BYTE   0
05DB'  CD 0000:05       ..LIN1: CALL    ARG
05DE'  04040400         ..LINA: .BYTE   $IVAL,$IVAL,$IVAL,$TAF
05E2'  E5                       PUSH    H
05E3'  FD6606                   MOV     H,6(Y)  ; GET Y
```

```
05E6'  FD6E01            MOV     L,1(Y)  ; AND X
05E9'  ED5B 65EF         LDED    OLDXY   ; GET BACK OLD XY
05ED'  22 65EF           SHLD    OLDXY   ; AND SET NEW
05F0'  FD7E0B            MOV     A,11(Y)
05F3'  A7                ANA     A
05F4'  2806              JRZ     ..NODR  ; YEP - SKIP DRAWING
05F6'  32 65EE           STA     WRMODE
05F9'  CD 0602'          CALL    DVECT
05FC'  E1          ..NODR: POP     H
05FD'  C9                RET
05FE'              ..LINX:
                   ; STRANGE TABLE ...
05FE'  0055AAFF    MSKTBL: .BYTE   0,055H,0AAH,0FFH
                   ; THIS ROUTINE IMPLEMENTS LARRY LIVERMORE'S VECTOR
                   ; DRAWING ALGORITHM.  SEE ZGRASS TECH DOC
                   ; INPUT:        L = X1 COORDINATE
                   ;               H = Y1 COORDINATE
                   ;               E = X2 COORDINATE
                   ;               D = Y2 COORDINATE
                   ;
                   ; OUTPUT:       A = 0, DE = X2,Y2, BC, HL CLOBBERED
                   ; RAM USE:
                   ; INCRO          2 BYTES HOLDS X,Y INCREMENTS
                   ; MNMX           2 BYTES HOLDS MIN, MAX DELTAS
                   ; PIXVAL         1 BYTE HOLDS LEFT JUSTIFIED PIXEL FOR XO
                   R WRITE
                   ; WRMODE         1 BYTE HOLDS PLOP-XOR FLAG
                   ; COMPUTE DELTAS AND ABS(DELTAS)
0602'  D5          DVECT:  PUSH    D
0603'  45                  MOV     B,L     ; COMPUTE Y STUFF
0604'  4B                  MOV     C,E
0605'  CD 0654'            CALL    CDELTA
0608'  58                  MOV     E,B
0609'  69                  MOV     L,C
060A'  44                  MOV     B,H
060B'  4A                  MOV     C,D     ; AND X STUFF
060C'  CD 0654'            CALL    CDELTA
060F'  61                  MOV     H,C
0610'  50                  MOV     D,B
0611'  22 65E9             SHLD    INCRO
                   ; DECIDE WHICH IS BIGGER - CALL BIGGER MX, SMALLER MN
0614'  0E00                MVI     C,0
0616'  7A                  MOV     A,D
0617'  BB                  CMP     E
0618'  3803                JRC     ..DV1
061A'  53                  MOV     D,E
061B'  5F                  MOV     E,A
061C'  0C                  INR     C
061D'  7A          ..DV1:  MOV     A,D
061E'  CB3F                SRLR    A
0620'  47                  MOV     B,A
0621'  EB                  XCHG
0622'  22 65EB             SHLD    MNMX
0625'  D1                  POP     D
0626'  7D                  MOV     A,L
```

```
0627'  3C                      INR     A
0628'  F5              VECT2:  PUSH    PSW
0629'  3A 65EE                 LDA     WRMODE
062C'  CD 00A0'                CALL    POINTR  ; DRAW THE POINT!
                       ; NOW INCREMENT COORDINATES
062F'  2A 65EB         VECT2A: LHLD    MNMX
0632'  78                      MOV     A,B
0633'  84                      ADD     H
0634'  BD                      CMP     L
0635'  380D                    JRC     VECT4   ; JUMP IF NOT
                       ; M+MN IS >=MX, SET M=MOD(M+MN,MX)
0637'  95                      SUB     L
0638'  47                      MOV     B,A
                       ; INCREMENT BOTH DIRECTIONS
0639'  2A 65E9                 LHLD    INCRO
063C'  7A                      MOV     A,D     ; CONFUSE Y
063D'  84                      ADD     H
063E'  57                      MOV     D,A
063F'  7B              VECT3:  MOV     A,E     ; THEN X
0640'  85                      ADD     L
0641'  5F                      MOV     E,A
0642'  180B                    JMPR    VECT5
                       ; M + MN IS < MX, SET M = M + MN
0644'  47              VECT4:  MOV     B,A
                       ; INCREMENT ONLY MAX DIMENSION
0645'  2A 65E9                 LHLD    INCRO
0648'  79                      MOV     A,C     ; C = DIRECTION FLAG
0649'  0F                      RRC
064A'  30F3                    JRNC    VECT3   ; 0=>X, SO GO DO IT
064C'  7A                      MOV     A,D     ; Y CASE
064D'  84                      ADD     H
064E'  57                      MOV     D,A
                       ; END OF LOOP
064F'  F1              VECT5:  POP     PSW
0650'  3D                      DCR     A
0651'  20D5                    JRNZ    VECT2
0653'  C9                      RET
                       ; SUBROUTINE TO COMPUTE DELTA AND INCREMENT FOR TWO COOR
                       DINATES
0654'  E5              CDELTA: PUSH    H
0655'  D5                      PUSH    D
0656'  69                      MOV     L,C
0657'  CD 0676'                CALL    SGNEXT
065A'  EB                      XCHG
065B'  68                      MOV     L,B
065C'  CD 0676'                CALL    SGNEXT
065F'  AF                      XRA     A
0660'  ED52                    DSBC    D
0662'  B4                      ORA     H
0663'  2807                    JRZ     ..CD1
0665'  4F                      MOV     C,A
0666'  7D                      MOV     A,L
0667'  2F                      CMA
0668'  3C                      INR     A
0669'  47                      MOV     B,A
```

```
066A'   1807            JMPR    ..CD3
066C'   B5      ..CD1:  ORA     L
066D'   2802            JRZ     ..CD2
066F'   3E01            MVI     A,1
0671'   45      ..CD2:  MOV     B,L
0672'   4F              MOV     C,A
0673'   D1      ..CD3:  POP     D
0674'   E1              POP     H
0675'   C9              RET
                ; SIGN EXTENSION SUBROUTINE
0676'   2600    SGNEXT: MVI     H,0
0678'   7D              MOV     A,L
0679'   A7              ANA     A
067A'   F0              RP
067B'   25              DCR     H
067C'   C9              RET
                ; ABSOLUTE VALUE ROUTINE
                ; THIS ROUTINE COMPUTES THE ABSOLUTE VALUE OF THE ARGUME
                NT
                ; PASSED IN A.  THE RESULT IS RETURNED IN A.
067D'   A7      ABS:    ANA     A
067E'   F0              RP
067F'   2F              CMA
0680'   3C              INR     A
0681'   C9              RET
                ; RELATIVE TO ABSOLUTE CONVERSION ROUTINE
0682'   7A      R2ACLP: MOV     A,D
0683'   CD 067D'        CALL    ABS
0686'   FE34            CPI     52
0688'   D0              RNC
0689'   7B              MOV     A,E
068A'   CD 067D'        CALL    ABS
068D'   FE51            CPI     81      ;
068F'   D0              RNC
                ; NONCLIPPING ENTRY POINT
0690'   C5      R2A:    PUSH    B
0691'   7A              MOV     A,D
0692'   2F              CMA
0693'   C634            ADI     52
0695'   6F              MOV     L,A
0696'   2600            MVI     H,0
0698'   29              DAD     H
0699'   29              DAD     H
069A'   29              DAD     H
069B'   44              MOV     B,H
069C'   4D              MOV     C,L
069D'   29              DAD     H
069E'   29              DAD     H
069F'   09              DAD     B
06A0'   7B              MOV     A,E
06A1'   C64F            ADI     79
06A3'   0F              RRC
06A4'   0F              RRC
06A5'   E63F            ANI     3FH
06A7'   4F              MOV     C,A
```

```
06A8'  0600              MVI     B,0
06AA'  09                DAD     B
06AB'  CBF4              SET     6,H
06AD'  7B                MOV     A,E
06AE'  3D                DCR     A
06AF'  E603              ANI     3
06B1'  37                STC
06B2'  C1                POP     B
06B3'  C9                RET
06B4'  C0          PIXTBL: .BYTE   11000000B
06B5'  30                .BYTE   110000B
06B6'  0C                .BYTE   1100B
06B7'  03                .BYTE   11B
                   .END
```

GRAPH -
+++++ SYMBOL TABLE +++++

```
AASN    005F        ABS     067D'       ALLOC   0000:04 X   ARG     0000:05 X
ARGSTK  60CC        AR.DIM  0009        AR.SIZ  0007        AR.TYP  0006
AUARTD  002A        AUARTS  002B        BACKGR  65CC        BAUDSE  0023
BLANK   0020        BOTRAM  6000        BOTTOM  64A9        BOX     041D'    I
BOXA    042E'       BOXS    042B'       BOXX    04F0'       BSAUD   0022
BX.MOD  0015        BX.X    0001        BX.XS   000B        BX.Y    0006
BX.YS   0010        BYTEPL  0028        CDELTA  0654'       CHARSL  65DD
CHARWI  0000:06 X   CIRCLE  00FD'       CLEAR   05A9'    I  CLEARS  60CA
CLIPPE  04F0'       CNTRL   000C        CNTRLC  65CD        CNTRLO  65D9
CNTRLU  0015        CNTRLZ  65E4        CPHLDE  0598'    I  CPLARE  611C
CPLSIZ  0140        CR      000D        CRAPPE  4FFF        CSBLOK  668D
CSFLAG  6260        CURCX   6814        CURCY   6816        CURREN  64A5
C.CO    0011        C.C1    0012        C.CX    000B        C.CY    000D
C.DP    0013        C.ST    0002        C.X     0003        C.XF    000F
C.XS    0007        C.Y     0005        C.YF    0010        C.YS    0009
DDTON   65CE        DEC5IY  0000:07 X   DEVBL   6589        DEVCL0  6579
DEVCL1  657B        DEVCL2  657D        DEVCL3  657F        DEVCL4  6581
DEVCL5  6583        DEVCL6  6585        DEVCL7  6587        DEVFB   65CB
DEVHCB  658F        DEVMO   658B        DEVNM   65B7        DEVNT   658D
DEVTNA  65BB        DEVTNB  65BF        DEVTNC  65C3        DEVVA   65BD
DEVVAR  6579        DEVVB   65C1        DEVVBL  6591        DEVVC   65C5
DEVVD   65C9        DEVVN   65B9        DEVVS   65C7        DEVXCD  65B3
DEVYCD  65B5        DISPLA  03AC'    I  DIV16   0000:08 X   DOBOX   0434'    I
DOLPLH  62E8        DOLPPT  62EA        DONCU   0000:09 X   DUMBST  6577
DVDE2   0000:0A X   DVECT   0602'       EDBCNT  64AD        EDCCNT  64A7
EDLONG  6812        EDMODE  681C        EDNAME  64A1        EDNCX   680C
EDNCY   680E        EDNEWS  64A5        EDOCX   6808        EDOCY   680A
EDPN    6806        EDPO    6804        EDPTRC  64AB        EDPTRL  64A9
EDSTR   681A        ERABIT  0002        ERRPGM  0000:0B X   ER.ARA  002F
ER.ARG  0034        ER.ASN  0015        ER.BOX  001A        ER.CHN  0002
ER.CMD  001F        ER.CNV  0016        ER.COR  001B        ER.CTL  0036
ER.DEL  0026        ER.DIM  0030        ER.DIV  0018        ER.DP   0037
ER.DSK  0019        ER.EDT  0035        ER.FMT  0038        ER.FNF  001C
ER.FOR  0028        ER.IMP  0003        ER.LAB  0025        ER.MAC  0022
ER.NAE  002D        ER.NAM  0029        ER.NEG  003A        ER.NOT  0023
ER.NUL  0039        ER.NUM  002B        ER.NXT  001D        ER.OFL  0017
ER.OPN  0014        ER.OVE  001E        ER.PAR  002A        ER.REN  002C
ER.RET  0024        ER.SEP  0021        ER.SNP  0031        ER.SPC  002E
ER.STK  0004        ER.SW   0032        ER.TER  0020        ER.UFL  0033
ER.UNF  0027        EXTDEL  002E        E.HVAL  0002        E.LVAL  0001
E.SIZ   0005        E.TYP   0000        E.VAL   0001        FCNTH   65D2
FCNTI   65D3        FCNTJ   65D4        FCNTK   65D5        FCNTL   65D6
FCNTV   65E0        FCNTY   65E3        FIRST   64A3        FLAGS   65CB
FOREGR  65D0        FRAGSI  0400        FREELS  65E5        FSTDOL  648C
FSTINT  62EC        FWDPTR  65E7        GPIXEL  0029'       GRCLRX  05CC'
HCAREA  6593        HORCB   0009        INC5IY  0000:0C X   INCRO   65E9
INFBK   000D        INLIN   000F        INMOD   000E        IPOP    0000:0D X
IPUSH   0000:0E X   JMUL    039D'       JUNK    6542        KBLOCK  65F1
KEYFLG  67FF        KEYPTK  6533        KEYTRK  67F7        LEDS    0024
LF      000A        LINE    05CC'    I  LISTON  65E2        LN      0000:0F X
MACSTU  6536        MACTOP  625E        MAGIC   000C        MAXFRG  0040
MNMX    65EB        MPBAV   0016        MPMO    0010        MPNCV   0015
MPNV    0017        MPTNA   0011        MPTNB   0012        MPTNC   0013
MPVIB   0014        MSKTBL  05FE'       M.POPS  0078        NBLKB   0000
NBLKM   0001        NCUCOM  0029        NCUDAT  0028        NEWBOT  6810
```

GRAPH -
+++++ SYMBOL TABLE +++++

```
NL      000A        NORML   0020        NSADDR  6802        NUMBUF  64A3
N.ACOS  0006        N.ARG   0003        N.ASIN  0005        N.ATAN  0007
N.CHSD  0034        N.CHSF  0015        N.CHSS  0074        N.COS   0003
N.DADD  002C        N.DDIV  002F        N.DIV   0004        N.DMUL  002E
N.DMUU  0036        N.DSUB  002D        N.EMAX  0017        N.EMIN  FF69
N.ESGN  0040        N.EXP   000A        N.FADD  0010        N.FDIV  0013
N.FIXD  001E        N.FIXS  001F        N.FLTD  001C        N.FLTS  001D
N.FMUL  0012        N.FSUB  0011        N.LN    0009        N.LOG   0008
N.MSGN  0080        N.NOP   0000        N.OFLO  0001        N.POPD  0038
N.POPF  0018        N.PTOD  0037        N.PTOF  0017        N.PTOS  0077
N.PUPI  001A        N.PWR   000B        N.SADD  006C        N.SDIV  006F
N.SIGN  0040        N.SIN   0002        N.SMUL  006E        N.SMUU  0076
N.SQRT  0001        N.SSUB  006D        N.TAN   0004        N.UFLO  0002
N.XCHD  0039        N.XCHF  0019        N.XCHS  0079        N.ZERO  0020
OLDCHR  64A0        OLDCUR  649F        OLDKEY  649D        OLDXY   65EF
ONEBUF  6729        OPRL    0014        OPRSP   655B        OPRSTK  6547
OPRSZ   655D        ORJOIN  00BC'       ORPT    00B6'       OUTOFF  62E7
PCNT    655E        PINPUT  0039'       PIXEL   0000'   I   PIXTBL  06B4'
PIXVAL  65ED        PLOP    00C7'       PLOP1   00C8'       PLOPNG  00EE'
POINT   007A'       POINTE  64A7        POINTR  00A0'       PONOFF  65DA
POUTPU  005A'       PRINTR  6540        PRIOR   00F0'       PRPLOP  00C3'
PUTSCN  01E9'       PUTTAP  0000:10 X   R2A     0690'   I   R2ACLP  0682'
RAMEND  7FFF        RAMSTR  6900        RANSHT  655F        RETNON  0000:11 X
RMDTMP  6538        RUBACK  0005        RUBOUT  007F        SAVESP  625C
SCLEAR  05B9'   I   SCNEED  0004        SCROLA  020E'       SCROLE  0214'   I
SCROLL  01FA'       SCROLS  020B'       SCROLX  0282'       SCRWIN  67CD
SGNEXT  0676'       SINE    0000:12 X   SNAP    0282'   I   SNP.DS  0016
SNP.XS  0012        SNP.YS  0014        SOPRSP  653D        SOPRSZ  653F
STACK   60C8        STAKTO  6000        STRSIZ  6800        SUBSTU  6534
TAB     0009        TAPBUF  64B3        TAPCON  64AF        TAPPRO  64B1
TBFEND  6533        TEMPHD  60CA        TEMPS   62E5        TMPARG  67AD
TOP     6818        TTYBEG  6261        TTYEND  62E1        TTYINT  62E3
TTYPTR  62E1        TXTWIN  67E2        UARTD   00E0        UARTFL  649C
UARTS   00E1        URINAL  0FFF        USREND  65F1        V3PTR   6573
VBLANK  000A        VCTR    0021        VDCHAR  649E        VDNLF   65CF
VECT2   0628'       VECT2A  062F'       VECT3   063F'       VECT4   0644'
VECT5   064F'       VIPLH   6545        VOICE0  6563        WRMODE  65EE
XORPT   00AE'       XPAND   0019        ZGIM2   0001        ZGREND  681D
$AADR   0010        $ADDRF  0007        $ADDRI  0005        $ADDRS  0009
$ANY    001A        $ANYNA  FFFC        $ANYVA  FFFE        $ARGPT  0011
$BGPTR  000F        $BNDL   0007        $CALLE  000D        $CMDAD  0018
$CPLBL  002A        $CSBLO  0028        $DATAP  0007        $DOLDE  0001
$DOLOO  0002        $DVAL   0000        $END    001C        $FADR   000E
$FLAGS  0002        $FORBL  0024        $FORPT  000B        $FVAL   0006
$GOSUB  001A        $IADR   000C        $INPBU  0018        $INPPT  0016
$IVAL   0004        $KEYBL  0026        $LENGT  0001        $LINPT  0005
$LOCPT  0009        $MIBBL  0022        $MIBEN  001B        $NAMAD  000A
$NAME   000A        $NASCI  0009        $NDEL   0080        $NLINK  0003
$NULL   0002        $NVALU  0005        $REPEA  001E        $RVSTU  0013
$SAME   0020        $SASCI  000A        $SLEN   0006        $STRAD  0008
$STRPT  0003        $TAF    0000        $TYPE   0000        $USE    0005
.BLNK.  0000:03 X   .DATA.  0000*   X   .PROG.  06B8'   X
```