ldr.ImageSystem

Commented Source
Source Format: MxAss (like Hypra-Ass)

```
.petscii
.include "godotlib.lib"

; --------------------------
;
;   ldr.ImageSystem
;   Loader for Image System multicolor pictures
;
;   1.00: 12.05.02, first release
;   1.01: 27.05.02, bug: color rams were exchanged - fixed
;
; --------------------------

;       .ob "l_images,p,w"
;
; --------------------------

              .ba $c000
;
; -------------------------- Equations

              .eq byte                  =$34
              .eq cnt                   =byte
              .eq gmode                 =$35
              .eq vr                    =$36
              .eq cr                    =$38
              .eq vv                    =$3a
              .eq status                =$90
              .eq offx                  =$b2
              .eq offy                  =offx+1
;
              .eq vrbuf                 =$c600
              .eq crbuf                 =vrbuf+1000
              .eq col0                  =vrbuf+2000
              .eq sprite                =$d015
;
; -------------------------- Header

              jmp start                 ; entry point

              .by $80                   ; this is a Loader
              .by $00                   ; with no specific requester
              .by $00                   ; and can be run from a RAM device
              .wo modend
              .by $00,$00               ; reserved bytes
              .tx "Image System    "
              .tx "1.01"
              .tx "27.05.02"
              .tx "A.Dettke       "
```

```
; -------------------------- Main
;
jerror      jmp error                ; when failed
;
start       jsr getname              ; first of all store image file name
;
stt2        jsr gd_xopen             ; open file
            jsr basin                ; skip start address ($3c00)
            bcs jerror
            jsr basin
            bcs jerror

; -------------------------- Get Multicolor Image

stt3        jsr gd_clrms             ; clear message bar

sk2         lda #7                   ; activity counter to 7 (controls gauge bar)
            sta cntwert              ; first retrieve multi mode colors
            sta $ff
            ldx #0                   ; out: "Colors 1"
            jsr tcopy

            lda #<(1024)             ; set counter for 1024 bytes (color RAM)
            sta ls_vekta8
            lda #>(1024)
            sta ls_vekta8+1
            lda #<(crbuf)
            sta sc_texttab
            lda #>(crbuf)
            sta sc_texttab+1
;
loop2       jsr basin                ; get color RAM (no conversion yet)
jerr3       bcs jerror
            sta (sc_texttab),y
            jsr action               ; activity (progression of status gauge bar)
            ldy #0
            inc sc_texttab
            bne sk31
            inc sc_texttab+1
sk31        lda ls_vekta8            ; count bytes
            bne sk41
            dec ls_vekta8+1
sk41        dec ls_vekta8
            lda ls_vekta8
            ora ls_vekta8+1
            beq sk42
            jmp loop2
;
; -------------------------

sk42        lda #50                  ; activity counter to 50
            sta cntwert
            sta $ff
            ldx #30                  ; out: "Bitmap"
```

```
                jsr tcopy
;
                lda #<(8191)            ; set counter to 8191 bytes (bitmap)
                sta ls_vekta8
                lda #>(8191)
                sta ls_vekta8+1
                ldx #0                  ; destination $4000
                ldy #$40
                stx sc_texttab
                sty sc_texttab+1
;
loop            jsr basin               ; get bitmap
jerr2           bcs jerr3
                sta byte
                jsr action              ; activity
                ldy #0
                ldx #4                  ; convert to 4bit indexes
bloop           lda #0
                asl byte
                rol
                asl byte
                rol
                sta (sc_texttab),y
                inc sc_texttab
                bne sk
                inc sc_texttab+1
sk              dex
                bne bloop
                lda ls_vekta8           ; count bytes
                bne sk1
                dec ls_vekta8+1
sk1             dec ls_vekta8
                lda ls_vekta8
                ora ls_vekta8+1
                beq sk21

                lda status              ; error while reading?
                beq loop
;
; --------------------------

sk21            jsr basin               ; get background color (byte 8192)
                bcs jerr2
                and #15                 ; isolate color value
                sta col0                ; store

; --------------------------

                lda #7                  ; activity counter back to 7 again
                sta cntwert
                sta $ff
                ldx #10                 ; out: "Colors 2"
                jsr tcopy

                lda #<(1000)            ; set counter to 1000 bytes (video RAM)
```

```
                sta ls_vekta8
                lda #>(1000)
                sta ls_vekta8+1
                lda #<(vrbuf)
                sta sc_texttab
                lda #>(vrbuf)
                sta sc_texttab+1
;
loop1           jsr basin               ; get color RAM (no conversion yet)
                bcs jerr2
                sta (sc_texttab),y
                jsr action              ; activity
                ldy #0
                inc sc_texttab
                bne sk3
                inc sc_texttab+1
sk3             lda ls_vekta8           ; count bytes
                bne sk4
                dec ls_vekta8+1
sk4             dec ls_vekta8
                lda ls_vekta8
                ora ls_vekta8+1
                beq sk5
                jmp loop1
;
; --------------------------

sk5             jsr gd_xclose           ; close file

; -------------------------- Convert to 4Bit

                lda #<(vrbuf)           ; set start addresses
                sta vr
                lda #>(vrbuf)
                sta vr+1
                lda #<(crbuf)
                sta cr
                lda #>(crbuf)
                sta cr+1
                lda #<($4000)
                sta sc_texttab
                lda #>($4000)
                sta sc_texttab+1
                lda #<(1000)            ; set counter to 1000
                sta ls_vekta8
                lda #>(1000)
                sta ls_vekta8+1
;
                lda #200                ; activity counter to 200
                sta cntwert
                sta $ff
                ldx #20                 ; out: “Convert”
                jsr tcopy
;
loop3           lda (vr),y              ; convert video RAM colors to 4bit
```

```
                pha
                lsr                     ; left pixel
                lsr
                lsr
                lsr
                sta col0+1
                pla                     ; right pixel
                and #$0f
                sta col0+2
                lda (cr),y              ; convert color RAM colors to 4bit
                and #$0f
                sta col0+3
                lda #32
                sta cnt
bloop1          jsr action              ; activity
                ldy #0
                lda (sc_texttab),y      ; get values from 4bit (0-3)
                tax
                lda col0,x              ; get color values from table
                tax
                lda dnib,x              ; get conversion values to GoDot from table
                sta (sc_texttab),y      ; write back to 4bit (double nibbles)
                inc sc_texttab          ; advance
                bne sk6
                inc sc_texttab+1
sk6             dec cnt                 ; one tile
                bne bloop1
                inc vr                  ; next tile
                bne sk7
                inc vr+1
sk7             inc cr
                bne sk8
                inc cr+1
sk8             lda ls_vekta8           ; 1000 tiles
                bne sk9
                dec ls_vekta8+1
sk9             dec ls_vekta8
                lda ls_vekta8
                ora ls_vekta8+1
                bne loop3

; -------------------------- Close File

sk11            ldx #30                 ; reset text to default ("Bitmap")
                jsr tcopy
                jsr setinfo             ; set filename for GoDot main screen info
;
sk10            jsr gd_xmess            ; error message from drive
                jsr gd_spron            ; sprite pointer on
                sec                     ; leave loader
                rts

; -------------------------- Error
error           jsr gd_xclose
                jsr sk10
```

```
              clc                       ; don't leave loader
              rts

; -------------------------- Activity Display

messout       ldx #<(message)
              ldy #>(message)
              jmp gd_xtxout2
;
tcopy         ldy #0
tc0           lda txt,x
              beq clrmess
              sta message,y
              inx
              iny
              bne tc0
;
action        dec $ff
              bne ld4
              lda cntwert
              sta $ff
              ldy offy
              ldx offx
              lda filltab,x
              sta mess,y
              jsr messout
              dec offx
              bpl ld4
              inc offy
              lda #7
              sta offx
ld4           rts
;
clrmess       ldx #20
              lda #32
cl0           sta mess,x
              dex
              bpl cl0
              ldy #0
              ldx #7
              sty offy
              stx offx
              rts
;
filltab       .by 160,93,103,127,126,124,105,109
;
cntwert       .by 50
;
txt           .ts " Colors 1@"          ; 0
              .ts " Colors 2@"          ; 10
              .ts " Convert @"          ; 20
              .ts " Bitmap  @"          ; 30
              .ts " Move    @"          ; 40            (unused)
;
message       .ts " Bitmap    "         ;               default text
```

```
mess        .tx "                                "          ; gauge bar
            .by 0

; --------------------------- Palette Table for MC double pixels
;
dnib        .by $00,$ff,$44,$cc,$55,$aa,$11,$dd
            .by $66,$22,$99,$33,$77,$ee,$88,$bb

; --------------------------- Save Image Filename
;
getname     ldx #0
si0         lda ls_lastname,x
            beq si1
            sta nbuf,x
            inx
            cpx #16
            bcc si0
si1         rts
;
; --------------------------- Set Image Information

setinfo     jsr setname
            jsr setloader
            jsr setmode
            jmp setdata
;
setname     lda #0
            ldx #<(ls_picname)
            ldy #>(ls_picname)
            bne si3
setloader   lda #17
            ldx #<(ls_iloader)
            ldy #>(ls_iloader)
            bne si3
setmode     lda #25
            ldx #<(ls_imode)
            ldy #>(ls_imode)
            bne si3
setdata     lda #33
            ldx #<(ls_idrive)
            ldy #>(ls_idrive)
si3         stx sc_texttab
            sty sc_texttab+1
            tax
            ldy #0
si4         lda nbuf,x
            beq si5
            sta (sc_texttab),y
            inx
            iny
            bne si4
si5         rts
;
nbuf        .by 32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,0
                                        ; image filename buffer
```

```
            .ts "ImgSyst@"          ; 17
modetx      .ts "160x200@"          ; 25
datatype    .ts "Color@"            ; 33

modend      .en
```