```
; Beispiel für einen Bildschirmschoner
; Das Beispiel wurde dem SourceCode von MegaPatch64
; entnommen und auf den C64 reduziert.
; Die vollständige Version inkl. der Anpassungen für den
; C128 kann dem MegaPatch SourceCode entnommen
; werden.

                        n "Starfield"
                        t "G3_SymMacExt"

                        a "M. Kanet"
                        f SYSTEM
                        o LD_ADDR_SCRSAVER

                        i
                                        

                        c "ScrSaver64  V1.0"
                        z $80   ;nur GEOS64 bei MP3-64

;*** ScreenSaver aufrufen.
:MainInit               jmp     InitScreenSaver

;*** ScreenSaver installieren.
;    Laufwerk von dem ScreenSaver geladen wurde muß noch aktiv sein!
;    Rückgabe eines Fehlers im xReg ($00=Kein Fehler).
;    ACHTUNG! Nur JMP-Befehl oder "LDX #$00:RTS", da direkt im Anschluß
;    der Name des ScreenSavers erwartet wird! (Addr: G3_ScrSave +6)
:InstallSaver           ldx     #$00
                        rts

;*** Name des ScreenSavers.
;    Direkt nach dem JMP-Befehl, da über den GEOS.Editor der Name
;    an dieser Stelle ausgelesen wird.
;    Der Name muss mit dem Dateinamen übereinstimmen, da der
;    Bildschirmschoner über diesen Namen beim Systemstart geladen wird.
:SaverName              b "Starfield",NULL
```

```
;*** ScreenSaver aufrufen.
:InitScreenSaver    php                 ;IRQ sperren.
                    sei                 ;Screener läuft in der MainLoop!

                    ldx     #$1f        ;Register ":r0" bis ":r3"
::51                lda     r0L,x       ;zwischenspeichern.
                    pha
                    dex
                    bpl     :51

                    jsr     DoSaverJob  ;Bildschirmschoner aktivieren.

                    lda     #%01000000  ;Bildschirmschoner neu starten.
                    sta     Flag_ScrSaver

                    ldx     #$00        ;Register ":r0" bis ":r3"
::52                pla                 ;zurückschreiben.
                    sta     r0L,x
                    inx
                    cpx     #$20
                    bne     :52

                    ldx     CPU_DATA    ;CPU-Register zwischenspeichern
und
                    lda     #$35        ;I/O-Bereich einblenden.
                    sta     CPU_DATA

::53                lda     #$00
                    sta     $dc00       ;Tastenregister aktivieren.
                    lda     $dc01       ;Tastenstatus einlesen.
                    eor     #$ff        ;Taste noch gedrückt ?
                    bne     :53         ;Ja, Warteschleife...

                    stx     CPU_DATA    ;CPU-Register zurücksetzen.

                    plp                 ;IRQ zurücksetzen und
                    rts                 ;Ende...
```

```
;*** Bildschirmschoner-Grafik.
:MaxStars           = 200
:DoSaverJob         PushB   $22         ;Register, die von RND(1)-Routine
                    PushB   $23         ;verändert werden, auf Stack
                    PushB   $26         ;zwischenspeichern.
                    PushB   $27
                    PushB   $28
                    PushB   $29
                    PushB   $56
                    PushB   $61
                    PushB   $62
                    PushB   $63
                    PushB   $64
                    PushB   $65
                    PushB   $66
                    PushB   $67
                    PushB   $68
                    PushB   $69
                    PushB   $6a
                    PushB   $6b
                    PushB   $6c
                    PushB   $6d
                    PushB   $6e
                    PushB   $6f
                    PushB   $70
                    PushB   $8b
                    PushB   $8c
                    PushB   $8d
                    PushB   $8e
                    PushB   $8f

                    ldx     CPU_DATA    ;IO-Bereich einblenden
                    lda     #$35
                    sta     CPU_DATA

                    lda     $d015       ;Sprites abschalten.
                    pha
                    lda     $d020
                    pha
                    lda     #$00
                    sta     $d015
                    sta     $d020

                    stx     CPU_DATA
                    jsr     PosScreenGrafx;Bildschirm-Inhalt retten.
                    jsr     StashRAM
                    jsr     PosScreenColor
                    jsr     StashRAM
```

```
                jsr     i_FillRam       ;Sternen-Farbe setzen.
                w       1000
                w       COLOR_MATRIX
                b       $10
                jsr     i_FillRam       ;Sternenhimmel löschen.
                w       8000
                w       SCREEN_BASE
                b       $00

::80            jsr     StarField

                jsr     PosScreenGrafx;Bildschirm-Inhalt zurücksetzen.
                jsr     FetchRAM
                jsr     PosScreenColor
                jsr     FetchRAM

                ldx     CPU_DATA        ;Sprites einschalten.
                lda     #$35
                sta     CPU_DATA

                pla                     ;Randfarbe VIC wiederherstellen
                sta     $d020
                pla
                sta     $d015

                stx     CPU_DATA        ;Register wieder zurücksetzen C64
                PopB    $8f
                PopB    $8e
                PopB    $8d
                PopB    $8c
                PopB    $8b
                PopB    $70
                PopB    $6f
                PopB    $6e
                PopB    $6d
                PopB    $6c
                PopB    $6b
                PopB    $6a
                PopB    $69
                PopB    $68
                PopB    $67
                PopB    $66
                PopB    $65
                PopB    $64
                PopB    $63
                PopB    $62
                PopB    $61
                PopB    $56
```

```
                    PopB    $29
                    PopB    $28
                    PopB    $27
                    PopB    $26
                    PopB    $23
                    PopB    $22
                    rts

;*** Bildschirm-Inhalt retten.
:PosScreenGrafx     LoadW   r0,SCREEN_BASE
                    LoadW   r1,R2_ADDR_SS_GRAFX
                    LoadW   r2,R2_SIZE_SS_GRAFX
                    lda     MP3_64K_SYSTEM
                    sta     r3L
                    rts

:PosScreenColor     LoadW   r0,COLOR_MATRIX
                    LoadW   r1,R2_ADDR_SS_COLOR
                    LoadW   r2,R2_SIZE_SS_COLOR
                    lda     MP3_64K_SYSTEM
                    sta     r3L
                    rts

;*** Tabelle mit Zufallszahlen erstellen.
;    Dazu wird intern die Routine RND(1) des BASIC-Interpreters
;    verwendet um Zufallszahlen im Register SEED ($008b-$008f) zu
;    erstellen. Dabei werden Zufallszahlen im Bereich 0-255 erstellt,
;    wobei jede Zahl nur 1x vorkommt.
:EditRandomTab      lda     CPU_DATA    ;CPU-Register speichern und
                    pha                 ;BASIC-Kernal einblenden.
                    lda     #$37
                    sta     CPU_DATA

                    lda     #%11001100  ;Startwert für RND-Funktion.
                    sta     $8b
                    eor     #%00110011
                    sta     $8c
                    eor     #%10101010
                    sta     $8d
                    eor     #%00011101
                    sta     $8e
                    eor     #%11100010
                    sta     $8f

                    ldy     #$00        ;Zeiger auf Tabelle löschen.

::51                tya                 ;Tabellenzeiger zwischenspeichern.
                    pha
```

```
::52                jsr     $e0be           ;RND(1)-Funktion aufrufen.

                    lda     $8e             ;Zufallszahl von 0-255 erstellen.
                    asl
                    eor     $8c
                    asl
                    eor     $8d
                    asl
                    eor     $8f

                    ldx     #$00            ;Zeiger auf Zahlentabelle.
::53                cmp     RandomTab,x     ;Ist Zahl bereits in Tabelle ?
                    bne     :54             ;Nein, weiter...
                    lda     RandomTab,x     ;Ist Zahl = $00 ?
                    bne     :52             ;Nein, neue Zahl suchen.
                    beq     :55             ;Ja, Zahl speichern.
::54                inx                     ;Zeiger auf nächste Zahl in Tabelle.
                    cpx     #MaxStars
                    bne     :53             ;Weitersuchen.

::55                tax                     ;Aktuelle Zufallszahl retten.
                    pla                     ;Zeiger auf Zahlentabelle wieder
                    tay                     ;in yReg kopieren.
                    txa                     ;Neue Zufallszahl in Zahlentabelle
                    sta     RandomTab,y     ;kopieren.
                    iny                     ;256 Zufallszahlen erstellt ?
                    cpy     #MaxStars
                    bne     :51             ;Nein, weiter...

                    pla                     ;CPU-Status zurücksetzen.
                    sta     CPU_DATA
                    rts

;*** Zufallszahl aus Tabelle einlesen.
;    Um die Zahlen nun ufälliger zu verteilen, Zahl mit
;    Rasterzeilen-Register verküpfen.
:InitRandom         lda     CPU_DATA        ;CPU-Register speichern und
                    pha                     ;I/O-Bereich einblenden.
                    lda     #$35
                    sta     CPU_DATA

                    ldx     r0L             ;Letzte Zufallszahl = $00 ?
                    bne     :51             ;Nein, weiter...
                    ldx     $d012           ;Rasterzeilen-Reigtsre als Zeiger.
```

```
::51            lda     RandomTab,x ;Zufallszahl aus Tabelle holen.
                eor     $d012       ;Mit rasterzeilen-Reg. verknüpfen.
                tax                 ;Als neuen Zeiger auf Tabelle ver-
                lda     RandomTab,x ;wenden und Zufallszahl
                sta     r0L         ;einlesen und zwischenspeichern.
                pla                 ;CPU-Status zurücksetzen.
                sta     CPU_DATA
                rts

;*** Sternenfeld zeichnen.
:StarField      jsr     EditRandomTab;Zufallszahlen erstellen.
                jsr     GetXYKoord  ;Startwerte für Sterne erstellen.

::51            ldy     #0          ;Zeiger auf ersten Stern.
::52            tya                 ;Sternzähler zwischenspeichern.
                pha
                jsr     MoveStar    ;Stern zeichnen und verschieben.
                pla                 ;Sternzähler zurücksetzen.
                tay

                ldx     CPU_DATA    ;CPU-Register speichern und
                lda     #$35        ;I/O-bereich einblenden.
                sta     CPU_DATA

                lda     $d012       ;Warteschleife.
::53            cmp     $d012
                beq     :53

                lda     #$00
                sta     $dc00       ;Tastenregister aktivieren.
                lda     $dc01       ;Tastenstatus einlesen.

                stx     CPU_DATA    ;CPU-Register zurücksetzen.

                eor     #$ff        ;Wurde Taste gedrückt ?
                bne     :54         ;Ja, Ende...

                iny                 ;Zeiger auf nächsten Stern.
                cpy     #150        ;Alle Sterne aufgebaut ?
                bne     :52         ;Nein, weiter...
                jmp     :51         ;Endloß-Schleife bis Taste gedrückt.
::54            rts

;*** Startwerte für alle Sterne berechnen.
:GetXYKoord     ldy     #$00
::51            jsr     SetStartKoord ;Startwerte für aktuellen Stern.
                iny                 ;Alle Sterne berechnet ?
                cpy     #MaxStars
                bne     :51         ;Nein, weiter...
                rts
```

```
;*** Startwerte für aktuellen Stern neu setzen.
:SetStartKoord      tya
                    and     #%00000111

                    clc
                    adc     #160 -4       ;X-Startposition von 160-167,
                    sta     Star_x_l,y    ;damit nicht alle Sterne am
                    lda     #0            ;gleichen Punkt beginnen.
::80                sta     Star_x_h,y
                    tya                   ;Y-Startposition von 100-103,
                    and     #%00000011    ;damit nicht alle Sterne am
                    clc                   ;gleichen Punkt beginnen.
                    adc     #100 -2
                    sta     Star_y,y

                    ldx     #%10001111    ;Zwangsrichtung bestimmen.
                    tya                   ;Um eine gleichmäßigere Verteilung
                    lsr                   ;der Sterne auf dem Bildschirm zu
                    bcc     :51           ;erreichen, wird jeder zweite Stern
                    ldx     #%10000011    ;extrem flach, bzw. extrem steil
::51                txa                   ;berechnet. Sonst erscheinen die
                    sta     r1L           ;Sterne in den Ecken des Bild-
                    eor     #%00001100    ;schirms konzentriert (X-Effekt).
                    sta     r1H

                    lda     DeltaX,y      ;Letzten Richtungswert einlesen und
                    sta     r0L           ;an Zufallszahlen-Routine geben.
                    jsr     InitRandom    ;Neuen Richtungswert bestimmen.

                    lda     r0L           ;Sternenrichtung und Geschwindig-
                    and     r1L           ;keit eingrenzen.
                    cpy     #$08          ;Sterne #8-#15 fast vertikal.
                    bcc     :52
                    cpy     #$10
                    bcs     :52
                    ora     #%01111111
::52                sta     DeltaX,y      ;Neuen Richtungswert speichern.
                    sta     DeltaXuse,y

                    lda     DeltaX,y      ;Letzten Richtungswert einlesen und
                    sta     r0L           ;an Zufallszahlen-Routine geben.
                    jsr     InitRandom    ;Neuen Richtungswert bestimmen.
```

```
                    lda     r0L             ;Sternenrichtung und Geschwindig-
                    and     r1H             ;keit eingrenzen.
                    cpy     #$08            ;Sterne #0-#7 fast horizontal.
                    bcs     :53
                    ora     #%01111111
::53                sta     DeltaY,y        ;Neuen Richtungswert speichern.
                    sta     DeltaYuse,y
                    rts

;*** Sternen-Koordinaten einlesen.
:SetStarKoord       lda     Star_x_l,y
                    sta     r3L
                    lda     Star_x_h,y
                    sta     r3H
                    lda     Star_y  ,y
                    sta     r11L
                    rts

;*** Stern verschieben.
:MoveStar           lda     DeltaXuse,y     ;Zähler für X-Richtung einlesen.
                    and     #%01111111      ;Neue X-Position setzen ?
                    beq     :51             ;Ja, weiter...

                    lda     DeltaXuse,y     ;Zähler für X-Richtung korrigieren.
                    and     #%10000000
                    sta     r0L
                    lda     DeltaXuse,y
                    and     #%01111111
                    sec
                    sbc     #$01
                    ora     r0L
                    sta     DeltaXuse,y
                    jmp     :56             ;Weiter mit Y-Richtung.

::51                jsr     ClrStar         ;Aktuellen Stern löschen.

                    lda     DeltaX,y        ;Zähler für X-Richtung neu
                    sta     DeltaXuse,y     ;initialisieren.
                    bmi     :52             ; => Stern fliegt in Gegenrichtung.

                    lda     Star_x_l,y      ;Stern nach rechts bewegen.
                    clc
                    adc     #$01
                    sta     Star_x_l,y
                    lda     Star_x_h,y
                    adc     #$00
                    sta     Star_x_h,y
                    jmp     :53
```

```
::52            lda     Star_x_l,y      ;Stern nach links bewegen.
                sec
                sbc     #$01
                sta     Star_x_l,y
                lda     Star_x_h,y
                sbc     #$00
                sta     Star_x_h,y
::53            lda     Star_x_l,y
                ora     Star_x_h,y      ;Hat Stern linken Rand erreicht ?
                beq     :55             ;Ja, neuen Stern berechnen.

                lda     Star_x_h,y
                cmp     #> 320
                bne     :54

                lda     Star_x_l,y
                cmp     #< 320          ;Hat Stern rechten Rand erreicht ?
::54            bne     :56             ;Nein, weiter.

::55            jsr     SetStartKoord   ;Neue Koordinate berechnen.

::56            lda     DeltaYuse,y     ;Zähler für Y-Richtung einlesen.
                and     #%01111111      ;Neue Y-Position setzen ?
                beq     :57             ;Ja, weiter...

                lda     DeltaYuse,y     ;Zähler für Y-Richtung korrigieren.
                and     #%10000000
                sta     r0L
                lda     DeltaYuse,y
                and     #%01111111
                sec
                sbc     #$01
                ora     r0L
                sta     DeltaYuse,y
                jmp     :61

::57            jsr     ClrStar         ;Aktuellen Stern löschen.

                lda     DeltaY,y        ;Zähler für Y-Richtung neu
                sta     DeltaYuse,y     ;initialisieren.
                bmi     :58             ; => Stern fliegt in Gegenrichtung.

                lda     Star_y,y        ;Stern nach unten bewegen.
                clc
                adc     #$01
                sta     Star_y,y
                jmp     :59
```

```
::58                lda     Star_y,y        ;Stern nach oben bewegen.
                    sec
                    sbc     #$01
                    sta     Star_y,y

::59                lda     Star_y,y        ;Hat Stern oberen Rand erreicht ?
                    beq     :60             ;Ja, neuen Stern berechnen.
                    cmp     #200            ;Hat Stern unteren Rand erreicht ?
                    bcc     :61             ;Nein, weiter...

::60                jsr     SetStarKoord    ;Neue Koordinate berechnen.

::61                jmp     DrawStar        ;Stern an neue Position einzeichnen.

;*** Stern-Pixel zeichnen.
;    Routine ist kompatibel zu DrawPoint ($C133).
;    Intern wird aber eine FastDrawPoint-Routine
;    zum schnelleren zeichnen verwendet.
:DrawStar           tya                     ;yReg zwischenspeichern.
                    pha
                    jsr     SetStarKoord    ;Sternen-Koordinaten einlesen.
;                   lda     #$00            ;Flag für DrawPoint setzen.
                    sec                     ;Flag für "Pixel setzen".
                    jsr     DrawPointXL     ;Pixel zeichnen.
                    pla                     ;yReg zurücksetzen.
                    tay
                    rts

;*** Stern-Pixel löschen.
;    Routine ist kompatibel zu DrawPoint ($C133).
;    Intern wird aber eine FastDrawPoint-Routine
;    zum schnelleren zeichnen verwendet.
:ClrStar            tya                     ;yReg zwischenspeichern.
                    pha
                    jsr     SetStarKoord    ;Sternen-Koordinaten einlesen.
;                   lda     #$00            ;Flag für DrawPoint setzen.
                    clc                     ;Flag für "Pixel löschen".
                    jsr     DrawPointXL     ;Pixel zeichnen.
                    pla                     ;yReg zurücksetzen.
                    tay
                    rts

;*** Schnelle ":DrawPoint"-Routine.
:DrawPointXL        php                     ;Pixel-Modus zwischenspeichern.

                    lda     r11L            ;Grafikzeile #0-#24 berechnen.
                    lsr
                    lsr
                    lsr
```

```
                tax
                lda     SCREEN_LINE_L,x
                sta     r2L
                lda     SCREEN_LINE_H,x
                sta     r2H

                lda     r3H             ;Spalte #0-#39 berechnen.
                lsr
                lda     r3L
                ror
                lsr
                lsr
                tax
                lda     SCREEN_COLUMN_L,x
                clc
                adc     r2L
                sta     r2L
                lda     SCREEN_COLUMN_H,x
                adc     r2H
                sta     r2H

                lda     r11L            ;Pixelzeile #0-#7 berechnen.
                and     #%00000111
                clc
                adc     r2L
                sta     r2L
                bcc     :51
                inc     r2H

::51            lda     r3L             ;Pixelspalte #0-#7 berechnen.
                and     #%00000111
                tax
                lda     SingleBitTab,x;Maske für aktuellen Pixel aus
                ldy     #$00            ;Tabelle einlesen.

                plp                     ;Pixel setzen/löschen ?
                bcc     :52             ; => löschen, weiter...
                ora     (r2L),y         ;Pixel setzen.
                sta     (r2L),y
                rts

::52            eor     #$ff            ;Pixel löschen.
                and     (r2L),y
                sta     (r2L),y
                rts
```

```
;*** Zwischenspeicher
:Star_x_l           s MaxStars
:Star_x_h           s MaxStars
:Star_y             s MaxStars
:DeltaX             s MaxStars
:DeltaY             s MaxStars
:DeltaXuse          s MaxStars
:DeltaYuse          s MaxStars
:RandomTab          s MaxStars

;*** Maskentabelle zum setzen/löschen von Bits.
:SingleBitTab       b $c0,$60,$30,$18,$0c,$06,$03,$03

;*** Startadressen der Grafikzeilen/LOW.
:SCREEN_LINE_L      b < SCREEN_BASE + 0*8*40
                    b < SCREEN_BASE + 1*8*40
                    b < SCREEN_BASE + 2*8*40
                    b < SCREEN_BASE + 3*8*40
                    b < SCREEN_BASE + 4*8*40
                    b < SCREEN_BASE + 5*8*40
                    b < SCREEN_BASE + 6*8*40
                    b < SCREEN_BASE + 7*8*40
                    b < SCREEN_BASE + 8*8*40
                    b < SCREEN_BASE + 9*8*40
                    b < SCREEN_BASE +10*8*40
                    b < SCREEN_BASE +11*8*40
                    b < SCREEN_BASE +12*8*40
                    b < SCREEN_BASE +13*8*40
                    b < SCREEN_BASE +14*8*40
                    b < SCREEN_BASE +15*8*40
                    b < SCREEN_BASE +16*8*40
                    b < SCREEN_BASE +17*8*40
                    b < SCREEN_BASE +18*8*40
                    b < SCREEN_BASE +19*8*40
                    b < SCREEN_BASE +20*8*40
                    b < SCREEN_BASE +21*8*40
                    b < SCREEN_BASE +22*8*40
                    b < SCREEN_BASE +23*8*40
                    b < SCREEN_BASE +24*8*40
```

```
;*** Startadressen der Grafikzeilen/HIGH.
:SCREEN_LINE_H      b > SCREEN_BASE + 0*8*40
                    b > SCREEN_BASE + 1*8*40
                    b > SCREEN_BASE + 2*8*40
                    b > SCREEN_BASE + 3*8*40
                    b > SCREEN_BASE + 4*8*40
                    b > SCREEN_BASE + 5*8*40
                    b > SCREEN_BASE + 6*8*40
                    b > SCREEN_BASE + 7*8*40
                    b > SCREEN_BASE + 8*8*40
                    b > SCREEN_BASE + 9*8*40
                    b > SCREEN_BASE +10*8*40
                    b > SCREEN_BASE +11*8*40
                    b > SCREEN_BASE +12*8*40
                    b > SCREEN_BASE +13*8*40
                    b > SCREEN_BASE +14*8*40
                    b > SCREEN_BASE +15*8*40
                    b > SCREEN_BASE +16*8*40
                    b > SCREEN_BASE +17*8*40
                    b > SCREEN_BASE +18*8*40
                    b > SCREEN_BASE +19*8*40
                    b > SCREEN_BASE +20*8*40
                    b > SCREEN_BASE +21*8*40
                    b > SCREEN_BASE +22*8*40
                    b > SCREEN_BASE +23*8*40
                    b > SCREEN_BASE +24*8*40
```

```
;*** Startadressen der Grafikspalten/LOW.
:SCREEN_COLUMN_L b < 8 * 0
                 b < 8 * 1
                 b < 8 * 2
                 b < 8 * 3
                 b < 8 * 4
                 b < 8 * 5
                 b < 8 * 6
                 b < 8 * 7
                 b < 8 * 8
                 b < 8 * 9
                 b < 8 * 10
                 b < 8 * 11
                 b < 8 * 12
                 b < 8 * 13
                 b < 8 * 14
                 b < 8 * 15
                 b < 8 * 16
                 b < 8 * 17
                 b < 8 * 18
                 b < 8 * 19
                 b < 8 * 20
                 b < 8 * 21
                 b < 8 * 22
                 b < 8 * 23
                 b < 8 * 24
                 b < 8 * 25
                 b < 8 * 26
                 b < 8 * 27
                 b < 8 * 28
                 b < 8 * 29
                 b < 8 * 30
                 b < 8 * 31
                 b < 8 * 32
                 b < 8 * 33
                 b < 8 * 34
                 b < 8 * 35
                 b < 8 * 36
                 b < 8 * 37
                 b < 8 * 38
                 b < 8 * 39
```

```
;*** Startadressen der Grafikspalten/HIGH.
:SCREEN_COLUMN_H b > 8 * 0
                 b > 8 * 1
                 b > 8 * 2
                 b > 8 * 3
                 b > 8 * 4
                 b > 8 * 5
                 b > 8 * 6
                 b > 8 * 7
                 b > 8 * 8
                 b > 8 * 9
                 b > 8 * 10
                 b > 8 * 11
                 b > 8 * 12
                 b > 8 * 13
                 b > 8 * 14
                 b > 8 * 15
                 b > 8 * 16
                 b > 8 * 17
                 b > 8 * 18
                 b > 8 * 19
                 b > 8 * 20
                 b > 8 * 21
                 b > 8 * 22
                 b > 8 * 23
                 b > 8 * 24
                 b > 8 * 25
                 b > 8 * 26
                 b > 8 * 27
                 b > 8 * 28
                 b > 8 * 29
                 b > 8 * 30
                 b > 8 * 31
                 b > 8 * 32
                 b > 8 * 33
                 b > 8 * 34
                 b > 8 * 35
                 b > 8 * 36
                 b > 8 * 37
                 b > 8 * 38
                 b > 8 * 39

;***************************************************************
;*** Endadresse testen.
;***************************************************************
                 g LD_ADDR_SCRSAVER + R2_SIZE_SCRSAVER -1
;***************************************************************
```