```
; ==============================================================
;  geoGopherUtl: utility routines
; ==============================================================
.if             Pass1
                .noeqin
                .include    geoGopherSym
                .include    geoGopherMac
                .include    geoGopher.inc
                .include    ultimate.inc
                .eqin
.endif
; ==============================================================
;  Check GEOS version number and show dialog if not version 2.0.
;               pass:       nothing
;               return:     carry set if < 2.0, clear otherwise
; ==============================================================
ckVersion:      lda         version
                tax
                and         #$f0
                lsr         a
                lsr         a
                lsr         a
                lsr         a
                ora         #$30
                sta         verHi               ;GEOS version advisory dialog
                sta         iVerHi              ;info dialog
                txa
                and         #$0f
                ora         #$30
                sta         verLo
                sta         iVerLo
                txa
                cmp         #$20
                beq         20$
                bcs         10$
                LoadW       r0,versDB           ;GEOS version <2.0
                LoadW       versMsg,loVers
                LoadW       RecoverVector,rstrDlg
                jsr         DoDlgBox
                sec
                rts
10$             LoadW       r0,versDB           ;GEOS version >2.0 (e.g. Wheels)
                LoadW       versMsg,hiVers
                LoadW       RecoverVector,rstrDlg
                jsr         DoDlgBox
20$             clc
                rts
```

```
; ==========================================================
;  Get track and sector addresses of VLIR modules
; ==========================================================
getMods:        LoadW      r6,fileName          ;get module load pointers
                LoadB      r7L,APPLICATION
                LoadB      r7H,1
                LoadW      r10,permName
                jsr        FindFTypes           ;we're looking for ourself
                txa
                beq        20$
10$             pha
                LoadW      a8,modsErr
                pla
                jsr        showCode
                jmp        EnterDeskTop
20$             LoadW      r0,fileName
                jsr        OpenRecordFile
                txa
                bne        10$
                LoadW      r0,$8104             ;fileHeader+4
                LoadW      r1,swapTS
                LoadW      r2,NUM_MODS*2
                jsr        MoveData
                jsr        CloseRecordFile
                rts
```

```
; ============================================================
;  Swap in a VLIR module using track and sector pointers. If the
;  specified module is already loaded, do nothing. Note that the font
;  module is auto-loaded (and is not stored as the current module).
;               pass:       .A, module number to load
;                           r7, address to load the module
;               return:     curMod, currently loaded module
;                           r7, end address of load +1
; ============================================================
swapMod:        cmp         curMod
                bne         10$
                rts
10$             sta         tempMod
                cmp         #MOD_FONT       ;gets auto-loaded
                beq         15$
                sta         curMod          ;VLIR module number
15$             sec
                sbc         #1
                asl         a
                tay
                lda         swapTS,y
                bne         20$
                lda         #2              ;invalid track: embedder not run?
                bne         60$
20$             sta         r1L
                lda         swapTS+1,y
                sta         r1H
                lda         #[$6000
                sec
                sbc         r7L
                sta         r2L
                lda         #]$6000
                sbc         r7H
                sta         r2H
                jsr         ReadFile
                txa
                bne         60$
                lda         tempMod
                cmp         #MOD_ULT        ;was network module just loaded?
                bne         40$             ;if so, auto-load font module
30$             jsr         j_init          ;set fontLoad (in network driver)
                MoveW       fontLoad,r7
                lda         #MOD_FONT
                bne         swapMod
40$             cmp         #MOD_FONT
                bne         50$
                lda         r7L             ;last byte loaded +1
                sta         itemsBuf        ;after network driver and font
                sta         itemsEnd
                lda         r7H
                sta         itemsBuf+1
                sta         itemsEnd+1
50$             rts
;               ================================================
60$             pha
                LoadW       a8,modErr
                pla
                jsr         showCode
                jmp         EnterDeskTop
```

```
; ============================================================
;  Show dialog box with error code and description.
;               pass:       .A, error code
;                           r8, address of descriptive message
; ============================================================
showCode:       jsr         byte2asc        ;error code in .A
                ldx         #0
10$             lda         ascNum,x
                beq         20$
                sta         errCode,x
                inx
                bne         10$
20$             lda         #' '
                sta         errCode,x
                inx
                ldy         #0
30$             lda         (a8),y
                sta         errCode,x
                beq         40$
                inx
                iny
                bne         30$
40$             LoadW       errMsg,badCode
                LoadW       r0,errorDB
                LoadW       RecoverVector,rstrDlg
                jsr         DoDlgBox
                rts
; ============================================================
;  Get string width in pixels.
;               pass:       string address in r0
;               return:     string length (in pixels) in a0
;               destroyed:  a1L
; ============================================================
strWidth:       ldy         #0
                sty         a0L
                sty         a0H
10$             lda         (r0),y
                beq         20$
                sty         a1L
                jsr         GetCharWidth
                clc
                adc         a0L
                sta         a0L
                lda         #0
                adc         a0H
                sta         a0H
                ldy         a1L
                iny
                bne         10$             ; string must be < 256 chars.
20$             rts
```

```
; ============================================================
;   Save or restore screen behind menus.
;               destroyed:  a8, a9
; ============================================================
saveMenu:   LoadW   a8,SCREEN_BASE
            LoadW   a9,menuSave
            LoadB   saveRstr,SCR_SAVE
            bne     doSavRst
rstrMenu:   LoadW   RecoverVector,RecoverRectangle ;restore vector
            LoadW   a8,menuSave
            LoadW   a9,SCREEN_BASE
            LoadB   saveRstr,SCR_RSTR
doSavRst:   ldy     #0
            sty     cards
            sty     cardRows
10$         ldx     #0
20$         lda     (a8),y
            sta     (a9),y
            iny
            bne     30$
            inc     a8H
            inc     a9H
30$         inx
            cpx     #8              ;one card
            bcc     20$
            inc     cards
            lda     cards
            cmp     #9              ;9 cards across
            bcc     10$
            LoadB   cards,0
            inc     cardRows
            lda     cardRows
            cmp     #6              ;6 card rows down
            bcs     50$
            lda     saveRstr
            cmp     #SCR_SAVE
            bne     40$
            AddVW   248,a8          ;320 - (8 * 9) thx White_Flame
            bra     10$
40$         AddVW   248,a9
            bra     10$
50$         rts
```

```
; ============================================================
;   Restore screen behind a dialog box by either redrawing the background
;  or re-rendering the gopher items.
; ============================================================
rstrDlg:        lda         numItems            ;need to redraw gopher items?
                bne         10$
                LoadB       r2L,32              ;standard dialog w/shadow
                LoadB       r2H,135
                LoadW       r3,64
                LoadW       r4,263
                lda         #2                  ;50% stipple
                jsr         SetPattern
                jsr         Rectangle
                bra         20$
10$             lda         itemType            ;trashed by doItems/showItem/mdText
                pha
                ldx         topItem
                jsr         doItems
                pla
                sta         itemType
20$             LoadW       RecoverVector,rstrDone  ;don't repeat for shadow
rstrDone:       rts
;               ================================================
rstrTDlg:       jsr         titleBar
                LoadW       r0,mainMenu
                php                             ;don't move mouse on DoMenu
                sei
                PushW       mouseXPos
                PushB       mouseYPos
                lda         #0
                jsr         DoMenu              ;redraw corrupted menu
                PopB        mouseYPos
                PopW        mouseXPos
                plp
                jsr         setGoph
                jsr         drawScrl
                jsr         doArrows
                LoadW       r0,0
                jsr         drawStat
                jsr         restItms            ;restore item state
                ldx         topItem
                jsr         doItems
                LoadW       RecoverVector,rstrDone  ;don't repeat for shadow
                rts
```

```
; ============================================================
;  Generic beep.
; ============================================================
beep:           php
                sei
                lda         $01
                pha
                and         #$f8
                ora         #$05
                sta         $01
                LoadB       $d400,#$31      ;voice 1 frequency low
                LoadB       $d401,#$1c      ;voice 1 frequency high
                LoadB       $d405,#$00      ;voice 1 attack/decay
                LoadB       $d406,#$f9      ;voice 1 sustain/release
                LoadB       $d418,#$0c      ;no filters, volume 15
                LoadB       $d404,#$11      ;gate on triangle, voice 1
                LoadB       $d404,#$10      ;gate off voice 1
                pla
                sta         $01
                plp
                rts
; ============================================================
;  Convert binary byte to decimal string by repeated subtraction.
;               pass:       .A, binary number
;               return:     null-terminated decimal string at ascNum
;               destroyed:  .Y
;                           a0L (minuend)
;                           a1L (accumulator)
;                           a1H (division constant)
; ============================================================
byte2asc:       sta         a0L
                ldy         #0
                sty         a1L
                lda         #100
                sta         a1H
10$             lda         a0L
20$             cmp         a1H
                bcc         30$
                sbc         a1H
                sta         a0L
                inc         a1L
                bne         20$
30$             lda         a1L
                bne         35$
                cpy         #0              ;no leading zeros
                beq         37$
35$             ora         #$30
                sta         ascNum,y
                iny
                lda         #0
                sta         a1L
37$             lda         a1H
                cmp         #10
                beq         40$
                lda         #10
                sta         a1H
                bne         10$
40$             lda         a0L
                ora         #$30
                sta         ascNum,y
                iny
                lda         #0
                sta         ascNum,y
                rts
```