Floating Point ROM Library#

General Information

Author: Optimized System Software OSS
Assembler: Mac65

```
;FPT8.M65
;
ptr1 = $8C
fr0  =  $D4    ; float reg 0
fr1  =  $E0    ; float reg 1
flptr  =  $FC    ; pointer to a fp num
cix    =$F2    ;index
inbuff  =  $F3    ; pointer to ascii num
intlbf =$DA51
lbuf = $580;
afp  =  $D800    ;ascii -> float
ifp  =  $D9AA    ; int in fr0 -> float in fr0
fpi  =  $D9D2    ; float in fr0 -> int in fr0
fasc  =  $D8E6    ; fr0 -> (inbuff)
fadd  =  $DA66    ; fr0 + fr1 -> fr0
fsub  =  $DA60    ; fr0 - fr1 -> fr0
fmul  =  $DADB    ; fr0 * fr1 -> fr0
fdiv  =  $DB28    ; fr0 / fr1 -> fr0
;
       .globl _atofp
;atofp(ascii,float)
_atofp:
;
       jsr     popax
       sta     flptr
       stx     flptr+1
       jsr     popax
       sta     inbuff
       stx     inbuff+1
       jsr     addeol
       lda     #0
       sta     cix
       jsr     afp
       jsr     store
       rts
;-----------------------------------
       .globl _itofp
;          itofp(int,float)
_itofp:
;
       jsr     popax
       sta     flptr
       stx     flptr+1
       jsr     popax
       sta     fr0
       stx     fr0+1
       jsr     ifp
       jsr     store
       rts
;-----------------------------------
       .globl _fptoi
;      int=fptoi(float)
```

```
_fptoi:
;
       jsr     popax
       jsr     loadfr0
       jsr     fpi
       lda     fr0
       ldx     fr0+1
       jsr     pushax
       rts
;------------------------------------

       .globl _fptoa
;      fptoa(ascii,float)
_fptoa:
       jsr     popax
       jsr     loadfr0
       jsr     popax
       sta     ptr1
       stx     ptr1+1
       lda     #0
       sta     cix
       jsr     intlbf
       jsr     fasc
       ldy     #$FF
       ldx     #$FF
loop2: inx
       lda     lbuf,x
       cmp     #$30
       beq     loop2
       dex
loop:  iny
       inx
       lda     lbuf,x
       sta     (ptr1),y
       bpl     loop
       and     #$7F
       sta     (ptr1),y
       iny
       lda     #0
       sta     (ptr1),y
       rts
;------------------------------------
       .globl _fpadd
;      fpadd(fp a,fp b,fp c)
;              a+b=c
_fpadd:
       jsr     getready
       jsr     fadd
       jsr     store
       rts
;------------------------------------
       .globl _fpsub
;      fpsub(fp a,fp b,fp c)
;              a-b=c
_fpsub:
       jsr     getready
       jsr     fsub
       jsr     store
       rts
```

```
;-----------------------------------
       .globl _fpmul
;      fpmul(fp a,fp b,fp c)
;              a*b=c
_fpmul:
       jsr     getready
       jsr     fmul
       jsr     store
       rts
;-----------------------------------
       .globl _fpdiv
;      fpdiv(fp a,fp b,fp c)
;              a/b=c
_fpdiv:
       jsr     getready
       jsr     fdiv
       jsr     store
       rts
;-----------------------------------
getready:
       jsr     popax
       sta     flptr
       stx     flptr+1
       jsr     popax
       jsr     loadfr1
       jsr     popax
       jsr     loadfr0
       rts
;-----------------------------------
store:
;
       ldx     #$05
       ldy     #$05
l1:    lda     fr0,x
       sta     (flptr),y
       dex
       dey
       bpl     l1
       rts
;-----------------------------------

loadfr0:
       sta     ptr1
       stx     ptr1+1
       ldx     #$05
       ldy     #$05
l2:    lda     (ptr1),y
       sta     fr0,x
       dex
       dey
       bpl     l2
       rts
;-----------------------------------
loadfr1:
       sta     ptr1
       stx     ptr1+1
       ldx     #$05
       ldy     #$05
l3:    lda     (ptr1),y
```

```
       sta     fr1,x
       dex
       dey
       bpl     l3
       rts
;-------------------------------------
addeol
       ldy     #$FF
l4:    iny
       lda     (inbuff),y
       bpl     l4
       lda     #$9B
       sta     (inbuff),y
       rts
;-------------------------------------
```