Sprite FPP

The basic theory behind a Sprite FPP is quite simple: Change sprite-pointer value every rasterline. Though there are a few things to prepare before it works in reality.

First we want to use a bigger graphics area than one sprite. With a normal $d017-stretcher we can make the sprites arbitrary high, and at the same time we only use one line of the sprite for graphics. If we then place our 8 sprites beside eachother, increasing the y-position by 1 for each sprite, then our FPP will display different graphics for each sprite. We have a 192 pixels wide sprite-layer with any desired height.

Note that all sprite-pointers have to be set each line for the FPP to work as we wish. Since they all use different lines of the sprite, the same value can be set to all sprite pointers. Using all 8 sprites, there will be 44 cycles availible each line for you to do this, and the $d017-stretching. Unrolled code is about the only option. There is no time for badlines here, so mostly you will need to use the $d011 idle-mode, which requires opening the upper/lower borders.

In the following code example i have kept it simple by using a loop for the FPP. Thus not all 8 sprites are being updated, only 6, so i save some cycles there to be able to loop the code. “FppCalc” is just a st0pid routine that generates some values to the FPP-table. Here you can do the real beauty of the effect by putting some more effort to it :).

```
    *= $0900

    sei
    lda #$ff ; Enable sprites
    sta $d015

    ldx #14 ; Set some sprite x-positions
    sec
    lda #$f0
    sta $d000,x
    sbc #$18
    dex
    dex
    bpl *-7

    ldx #14 ; Set some sprite y-positions
    clc
    lda #$40
    sta $d001,x
    adc #1
    dex
    dex
    bpl *-7

    lda #$bd ; Set idle-pattern
    sta $3fff
loop1
    lda #$f9 ; Remove upper/lower border
```

```
    cmp $d012
    bne *-3
    lda #0
    sta $d011
    bit $d011
    bpl *-3
    lda #8 ; Set $d011 to idle mode -> no badlines
    sta $d011

    jsr FppCalc ; Make beautiful FPP.

    lda #$48 ; Wait for sprite y-position
    cmp $d012
    bne *-3

    ldx #5 ; Wait a few cycles to make the d017-stretch work..
    dex    ; ..and all sprite pointers update on the same line.
    bne *-1
    nop

    ldx #0
loop2
    dec $d017 ; Do $d017-stretching as we have learned
    lda FppTab,x ; Get value from table..
    sta $07f8    ; ..and store to all sprite pointers
    sta $07f9
    sta $07fa
    sta $07fb
    sta $07fc
    sta $07fd

    ; Note: here we should store to $07fe and $07ff also,
    ; but there are not enough cycles in this loop (44 cycles).
    ; To make FPP on all sprites, unroll the loop.

    inc $d017 ; Set back $d017
    inx
    bpl loop2

    jmp loop1

FppCalc
    lda #0 ; Increase adding value
    inc *-1
    sta AddVal

    clc
    lda #0 ; Increase starting value
    adc #3
    sta *-3
    sta Value
```

```
    ldy #0
FC_1
    clc ; Calculate some obscure values to the FppTable
    lda AddVal
    adc #3
    sta AddVal
    bpl *+4
    eor #$ff
    lsr
    lsr
    lsr
    lsr
    lsr
    clc
    adc Value
    sta Value
    sta FppTab,y
    iny
    bpl FC_1
    rts

AddVal  .byte 0
Value   .byte 0

    .align $100 ; Align the table to a new page, this way lda FppTab,x
always takes 4 cycles.
FppTab
    .dsb 256 ; Reserve 256 bytes for the table
```