Stable Raster with Lightpen

I've seen a few references to getting a stable raster by triggering the light pen but couldn't find it in any tutorials. So I was curious about it and have documented here what I've found out.

It turns out to be a mostly useless technique because of the following problems:

- It's wrecked if the user presses space or joystick #1 button
- It can only be used once per frame!

But if you can live with that (for example if you check for space/button anyway to exit an intro) maybe you could use it. And you might **want** to use it because it has one big advantage:

You can have a stable raster in less than one raster line, using only one IRQ!

You don't have to set up a stable raster, then set timers, then use them in future IRQs - just one and you're done. Also, you don't need sprites or any other DMA.

How does it work?

It's based on the fact that you can manually trigger the light pen and read off the raster X coordinate from it, and if you do it in your IRQ routine the X coordinate will correspond to the amount your IRQ was delayed.

Here's a step-by-step explanation but see the Vic-II doc by Christian Bauer for more technical details:

1. Connect the light pen to port B of CIA A
2. In your IRQ routine, trigger light pen by writing $ff then $00 to $dc01
3. Read light pen X coordinate from $d013
4. Convert this to a value from 0 to 7
5. Use value in a clockslide

Code Example

The following code can get a stable raster in 45 cycles but it wastes 256 bytes on a table for converting the LP X value to the clockslide value. If you set “use_table = false” at the top to do the calculation manually, it will not need the table but will take 49 cycles.

The code shows the jitter value at the top right of the screen by incrementing a character corresponding to each jitter value, from 0 to 7.

```
; code should compile with 64tass V1.53.1590 or thereabouts

; 6502 w/ illegal instructions
.cpu "6502i"

; ---- config ------------------------------------------------------------
```

```
---

irq_vec     = $fffe
nmi_vec     = $fffa
screen      = $400
r_interrupt = 30
use_table   = true

; ---- code ----------------------------------------------------------------
---

* = $0801

        ; basic stub
        .word endbasic                  ; link to next line
        .word 2018                      ; line number
        .null $9E, format("%d", start)  ; SYS token
    endbasic:
        .byte 0,0,0                     ; end of BASIC program

    start:
        ; paranoia
        cld

        ; disable interrupts
        sei

        ; swap out basic and kernel roms
        lda #%00101111
        sta $00
        lda #%00111101
        sta $01

        ; disable timer interrupts
        lda #$7f
        sta $dc0d
        sta $dd0d

        ; lock out NMI interrupts
        lda #<nmi
        sta nmi_vec
        lda #>nmi
        sta nmi_vec+1
        lda #$00
        sta $dd0e           ; stop timer A
        sta $dd04           ; zero timer A
        sta $dd05
        lda #%10000001
        sta $dd0d           ; enable timer A NMI
        lda #$01
        sta $dd0e           ; start timer A, causing immediate NMI
```

```
        nop                 ; some delay for paranoia sake
        nop
        jmp clear_screen

    nmi:
        rti

    clear_screen:
        ldx #$00
    -   lda #$20
        sta $0400,x
        sta $0500,x
        sta $0600,x
        sta $0700,x
        lda #$01
        sta $d800,x
        sta $d900,x
        sta $da00,x
        lda #$06
        sta $db00,x
        inx
        bne -

        ; text
        scrpos = screen + ( 1*40) + 32
        ldx #textlen
    -   lda text,x
        sta scrpos,x
        dex
        bpl -

        ; init lightpen
        lda #%00010000
        sta $dc03
        lda #$ff
        sta $dc01

        ; set the interrupt vector
        lda #<irq1
        ldx #>irq1
        sta irq_vec
        stx irq_vec+1
        ; set the initial raster interrupt location
        lda #(r_interrupt & $ff)
        sta $d012
        lda $d011
        .if r_interrupt < $100
            and #%01111111
        .else
            ora #%10000000
        .endif
```

```
        sta $d011
        ; enable raster interrupt
        lda #$01
        sta $d01a
        ; ack video interrupts
        lsr $d019

        ; let the interrupts begin
        cli

    ; this mainloop ensures every possible jitter value is hit at some point
    mainloop:
        inc $700,x
        bne mainloop
        inx
        jmp mainloop

; ---- irq ---------------------------------------------------------------
---

.option allow_branch_across_page = 0

irq1:
        ; CPU takes 9 to 16 cycles

        ; skip saving registers, assume real mainloop would be simply jmp *

        ; ack (6 cycles)
        lsr $d019

        ; get raster X by triggering lightpen (6 cycles)
        lda #$00
        sta $dc01

        .if use_table
            ; use 256 byte table to shave 4 cycles off manual method (8
cycles)
            ldy $d013
            lda table_delay,y

        .else
            ; manually calc jitter value (12 cycles)
            lda $d013
            lsr
            lsr
            sbc #$08
            and #%00000111
        .endif

        ; clockslide (16 to 9 cycles)
        sta mod_clockslide0
```

```
    mod_clockslide0 = *+1
        bpl *
        lda #$a9
        lda #$a9
        lda #$ad
        lda $ea

        ; ---- stable here (45 cycles w/ table, 49 cycles without) ----

        ; border
        inc $d020

        ; delay for about 10 raster lines
        ldy #$83
    -   iny
        bne -

        ; now delay a bit more, increasing delay by one cycle every few
frames
        ; and restart after 31 increases
        dec cntr
        lda cntr
        lsr
        lsr
        lsr
        and #$1f        ; delay from 0 to 31

        sta mod_clockslide1
    mod_clockslide1 = *+1
        bpl *
        lda #$a9        ; 0
        lda #$a9        ; 2
        lda #$a9
        lda #$a9
        lda #$a9
        lda #$a9        ; 10
        lda #$a9
        lda #$a9
        lda #$a9
        lda #$a9
        lda #$a9        ; 20
        lda #$a9
        lda #$a9
        lda #$a9
        lda #$ad
        lda $ea         ; 30

        ; border
        dec $d020

        ; display lightpen x
```

```
        ldx mod_clockslide0
        inc screen + 32,x

endirq:
        ; reset lightpen
        lda #$ff
        sta $dc01

        rti


; ---- data ----------------------------------------------------------------
---

cntr:   .byte 00

text:   .text "01234567"
textlen = * - text - 1

.if use_table
    ; entire page of delay values; we only really need about 4*7 values in
the table, but
    ; because of spurious lightpen readings we need to have all the other
values covered
    ; to prevent a crash
    table_delay:
            .fill $20, 0

            .page   ; no page crossing here to ensure no extra cycles
            .fill $04, 0
            .fill $04, 1
            .fill $04, 2
            .fill $04, 3
            .fill $04, 4
            .fill $04, 5
            .fill $04, 6
            .fill $04, 7
            .endp

            .fill $c0, 7
.endif
```

Download Example

Here's a zipped prg file compiled from the above:

lp-stable-raster.zip