```
;=================================================================
==================================
;
RASTER INTERRUPTS
;=================================================================
==================================
;
Peter 'Sig' Hewett
;
- 2016
; A chain of raster irq's and routines for installing/removing
them
;-----------------------------------------------------------------
---------------------------------
;-----------------------------------------------------------------
---------------------------------
;
INSTALL RASTER IRQ
;-----------------------------------------------------------------
---------------------------------

InitRasterIRQ
        sei                     ; stop all interrupts
        lda PROC_PORT

        lda #$7f                ; disable cia #1 generating
timer irqs
        sta INT_CONTROL         ; which are used by the system
to flash cursor, etc.

        lda #$01                ; tell the VIC we want to
generate raster irqs
                                ; Note - by directly writing
#$01 and not setting bits
                                ; we are also turning off
sprite/sprite sprite/background
                                ; and light pen interrupts.. But
those are rather shite anyways
                                ; and won't be missed

        sta VIC_INTERRUPT_CONTROL

        lda #$10                ; number of the rasterline we
want the IRQ to occur at
        sta VIC_RASTER_LINE     ; we used this for WaitFrame,
remember? Reading gives the current
```

```
                                ; raster line, writing sets the
line for a raster interrupt to occur

                                ; The raster counter goes from
0-312, so we need to set the
                                ; most significant bit (MSB)

                                ; The MSB for setting the raster
line is bit 7 of $D011 - this
                                ; is also VIC_SCREEN_CONTROL,
that sets the height of the screen,
                                ; if it's turned on, text or
bitmap mode, and other things.
                                ; so we could easily use this
'short' method
;        lda #$14               ; screen on and at 25 rows..
;        sta $D011

                                ; But doing things properly and
only setting the bits we want is a
                                ; good practice to get into.
                                ; also we now have the option of
turning the screen on when we want
                                ; to - like after everything is
set up

        lda VIC_SCREEN_CONTROL  ; Fetch the VIC_SCREEN_CONTROL
        and #%01111111          ; mask the surrounding bits
;        ora #%00000000          ; or in the value we want to
set the MSB (Most significant bit)
                                ; in this case, it's cleared
        sta VIC_SCREEN_CONTROL
                                ; set the irq vector to point to
our routine
        lda #<IrqTopScreen
        sta $0314
        lda #>IrqTopScreen
        sta $0315
                                ; Acknowlege any pending cia
timer interrupts
                                ; just to be 100% safe
        lda $dc0d
        lda $dd0d

        cli                     ; turn interrupts back on
        rts
```

```
;-------------------------------------------------------------------
-----------------------------------
;
RELEASE RASTER IRQs
;-------------------------------------------------------------------
-----------------------------------
ReleaseRasterIRQ
        sei                             ; stop all interrupts

        lda #$37                        ; make sure the IO regs
at $dxxx
        sta $01                         ; are visible

        lda #$ff                        ; enable cia #1
generating timing irq's
        sta $dc0d

        lda #$00                        ; no more raster IRQ's
        sta $d01a

        lda #$31
        sta $0314
        lda #$ea
        sta $0315

        lda $dc0d                       ; acknowlege any pending
cia timer interrupts
        lda $dd0d

        cli
        rts

;===================================================================
==================================
;
IRQ - TOP SCREEN
;===================================================================
==================================
; Irq set to the very top of the visible screen (IN BORDER) good
for screen setup and timers
;
;-------------------------------------------------------------------
-----------------------------------
#region "IrqTopScreen"
IrqTopScreen
        sei                     ; acknowledge VIC irq
```

```
        lda $D019
        sta $D019
                                ; install glitch irq
        lda #<IrqGlitchCatcher
        sta $0314
        lda #>IrqGlitchCatcher
        sta $0315

        lda #$BF
        sta $D012


;=================================================================
======== OUR CODE GOES HERE
@start
        lda #COLOR_BLUE
        sta VIC_BORDER_COLOR

        lda CURRENT_SCREEN + 1          ; Hi byte of the current
screen
        cmp #>SCREEN2_MEM               ; compare to start of
Screen2
        beq @screen2

        lda #%00000010                  ; Set VIC to Screen0,
Charset 1
        sta VIC_MEMORY_CONTROL
        jmp @scroll

@screen2
        lda #%00010010                  ; Set VIC to Screen1,
Charset 1
        sta VIC_MEMORY_CONTROL

@scroll
        ;-------------------------------------------------------
SCREEN HARDWARE SCROLL (VERT)
        lda VIC_SCREEN_CONTROL_Y        ; Take the current
values
        and #%11111000                  ; mask out the scroll
values
        ora SCROLL_COUNT_Y              ; or in the scroll count
(bits 0-2 - y scroll value)
        sta VIC_SCREEN_CONTROL_Y        ; save the updated info
in the registers
```

```
        ;-------------------------------------------------------
SCREEN HARDWARE SCROLL (HORIZ)
        lda VIC_SCREEN_CONTROL_X        ; Take the current
values (Set at IrqScoreBoard)
        and #%11111000                  ; mask out the lower 4
bits (screen cols and scroll)
        ora SCROLL_COUNT_X              ; Or in the scroll count
(bits 0-2 - x scroll value)
        sta VIC_SCREEN_CONTROL_X        ; Save the updated info

        ;-------------------------------------------------------
TIMERS AND SYSTEM UPDATES
 ;       jsr UpdateTImers                   ; Update our timers
and automatic systems
        jsr ReadJoystick
        jsr JoyButton

        lda #COLOR_VIOLET
        sta VIC_BORDER_COLOR

        ;-------------------------------------------------------
-----------------------------------
        cli
        jmp $ea31

        rts
#endregion
;=============================================================
====================================
;
IRQ GLITCHCATHER
;=============================================================
====================================
; Force badline, black out character garble from scroll, and
adjust timing glitches
;-------------------------------------------------------------
-----------------------------------
#region "IRQGlitchCatcher"
IrqGlitchCatcher
        sei
                                ; acknowledge VIC irq
        lda $D019
        sta $D019

                                ; install scroller irq
        lda #<IrqScoreBoard
        sta $0314
```

```
        lda #>IrqScoreBoard
        sta $0315

                                ; nr of rasterline we want the
NEXT irq to occur at
        lda #$c7                ; Scoreboard appears 8 raster
lines after the glitch catcher
        sta $D012
        ;-------------------------------------------------------
-----------------------------------

        lda SCROLL_COUNT_Y
        beq @fr_0
        cmp #7
        beq @fr_7
        cmp #1
        beq @fr_1
        cmp #2
        beq @fr_2
        cmp #3
        beq @fr_3
        cmp #4
        beq @fr_4
        cmp #5
        beq @fr_5
        cmp #6
        beq @fr_6

        jmp @start

@fr_1
      ; nop
@fr_2
        nop
        nop
@fr_3
        nop
        nop
        nop
@fr_4
        nop
@fr_5
        nop
        nop
@fr_6
        nop
        nop
```

```
        nop
        nop
        nop
@fr_7
        nop
@fr_0

@start



        lda #%01100100                          ; Set VIC to
Screen 6, Charset 2
        sta VIC_MEMORY_CONTROL
        lda #%01010111                          ; Set Y to
scroll 7 to force badline to every frame
        sta VIC_SCREEN_CONTROL_Y                ; set extended
background mode
        lda #%11010000
        sta VIC_SCREEN_CONTROL_X                ; X scroll to 0
/ multicolor on / 38 cols
                                                ; If you set
multicolor AND extended background
                                                ; you get an
illegal mode that sets everything to
                                                ; black
        lda #COLOR_BLACK
        sta VIC_BACKGROUND_COLOR
        sta VIC_BORDER_COLOR
        ;-------------------------------------------------------
-----------------------------------
        cli
        jmp $ea31
#endregion
;================================================================
==================================
;
IRQ - BOTTOM SCREEN / SCOREBOARD
;================================================================
==================================
; IRQ at the top of the scoreboard

IrqScoreBoard
        sei                         ; acknowledge VIC irq
        lda $D019
        sta $D019
```

```
                                ; install scroller irq
        lda #<IrqTopScreen
        sta $0314
        lda #>IrqTopScreen
        sta $0315

                                 ; nr of rasterline we want the
NEXT irq to occur at
        lda #$10
        sta $D012

;===============================================================
======== OUR CODE GOES HERE
;        lda #%01100100                  ; Set VIC to
Screen 6, Charset 2
;        sta VIC_MEMORY_CONTROL

        lda #COLOR_GREEN
        sta VIC_BORDER_COLOR

        lda #%00010000                  ; Restore to Y
Scroll = 0
        sta VIC_SCREEN_CONTROL_Y        ; Be aware that
:
                                        ; bit #0-2 =
vertical scroll
                                        ; bit #3 =
screen height (0 = 24 rows)
                                        ; bit #4 =
screen on/off
                                        ; bit #5 =
text/bitmap (0 = text)
                                        ; bit #6 =
extended background on/off
                                        ; bit #7 =
read/write current raster line bit #8
                                        ; So '3' is the
default vert scroll location
        lda #%11010000
        sta VIC_SCREEN_CONTROL_X        ; Set screen to
default
                                        ; bit #3
controls screen width (0 = 38 cols)
                                        ; bit #4
controls multicolor (0 = off)
```

```
        ;-------------------------------------------------------------
-----------------------------------
        cli
        jmp $ea31

        rts
```