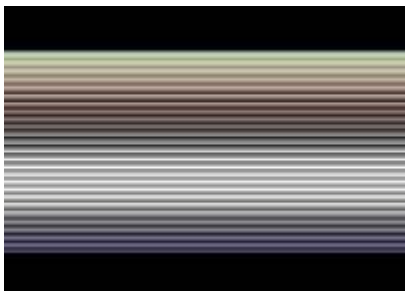
Rasterbar Flash Screen Effect

A Rasterbar Screen Flasher routine by Wozza/CygnusOz ~ you can assemble via Kick Assembler!



rasterbar-flasher_src.rar

You can achieve this effect in a number of ways, with obvious tighter code etc, but this is my poor version. When I first started coding back in the early 80's, Rasterbars were a lot of fun, but I remember nutting out a flash routine which I thought was very cool. I didn't do much coding after that, (too busy swapping, chasing girls and boozing) so after 20+ years (ahem) it's been fun to go back and re-visit.

If you are learning to code basic ASM routines then this is a good start; you can get different effects by changing a few bytes. Muck around and play! Try adding in another IRQ for say, a scroller etc...

```
// autostart
.pc = $0801 "Basic Program Start"
:BasicUpstart(start)
```

```
//set variables and locations
.var colours    =    $0a00
.var timing =    $0b00
.var pattern    =     $0c00
```

```
.pc = $0810 "Main Program"
start:  cld
    sei                 // setup a simple irq
    jsr $e544
    ldx #>irq1
    stx $0315
    ldx #<irq1
    stx $0314
    lda #$01
    sta $d019
    sta $d01a
    lda #$0d
    sta $dc0d
    cli
infin:  jmp infin
```

```
irq1:   asl $d019
    lda $13
    sta $d016
    ldx #$3a         //set raster
!loop:  cpx $d012
    bne !loop-
    ldx #$08
!loop:  dex
    bpl !loop-
    nop               // waste some cycles.
    nop
    nop
    ldx #$00
colrout: lda colours,x  //load in our custom colours
    sta $d020
    sta $d021
    ldy timing,x    //and corresponding timing.
!loop:  dey
    bpl !loop-
    inx
    cpx #$c0          //how many do we have? - 192
    bne colrout     //keep going otherwise..
    lda #$00        //clear
    sta $d020
    sta $d021
    ldx #$00
!loop:   lda colours+$01,x     //store our colours
    sta colours,x
    inx
    cpx #$c0
    bne !loop-
aa: lda pattern+$df       // =)
    sta colours+$08
ab: lda pattern+$e0       //+1
    sta colours+$10       //+8
ac: lda pattern+$e1       //+2
    sta colours+$18       //+8
ad: lda pattern+$e2       //+3
    sta colours+$20       //+8
ae: lda pattern+$e3       //etc
    sta colours+$28
af: lda pattern+$e4
    sta colours+$30
ag: lda pattern+$e5
    sta colours+$38
ah: lda pattern+$e6
    sta colours+$40
ai: lda pattern+$e7
    sta colours+$48
aj: lda pattern+$e8
```

```
    sta colours+$50
ak: lda pattern+$e9
    sta colours+$58
al: lda pattern+$ea
    sta colours+$60
am: lda pattern+$eb
    sta colours+$68
an: lda pattern+$ec
    sta colours+$70
ao: lda pattern+$ed
    sta colours+$78
ap: lda pattern+$ee
    sta colours+$80
aq: lda pattern+$ef
    sta colours+$88
ar: lda pattern+$f0
    sta colours+$90
as: lda pattern+$f1
    sta colours+$98
at: lda pattern+$f2
    sta colours+$a0
au: lda pattern+$f3
    sta colours+$a8
av: lda pattern+$f4
    sta colours+$b0
ax: lda pattern+$f5
    sta colours+$b8
ay: lda pattern+$f6
    sta colours+$c0
az: lda pattern+$18
    sta colours+$c8
    inc aa+$01          //increase count of each
    inc ab+$01          //line
    inc ac+$01
    inc ad+$01
    inc ae+$01
    inc af+$01
    inc ag+$01
    inc ah+$01
    inc ai+$01
    inc aj+$01
    inc ak+$01
    inc al+$01
    inc am+$01
    inc an+$01
    inc ao+$01
    inc ap+$01
    inc aq+$01
    inc ar+$01
    inc as+$01
    inc at+$01
```

```
    inc au+$01
    inc av+$01
    inc ax+$01
    inc ay+$01
    inc az+$01
    jmp $ea31
```

```
.pc = colours "colours"
.byte $0f,$01,$01,$0f,$0f,$0c,$0c,$0b
.byte $0b,$01,$0f,$0f,$0c,$0c,$0b,$0b
.byte $00,$0f,$0f,$0c,$0c,$0b,$0b,$00
.byte $06,$0f,$0c,$0c,$0b,$0b,$00,$06
.byte $06,$0c,$0c,$0b,$0b,$00,$06,$06
.byte $0e,$0c,$0b,$0b,$00,$06,$06,$08
.byte $0e,$0b,$0b,$00,$06,$06,$0e,$0e
.byte $03,$0b,$00,$06,$06,$0e,$0e,$03
.byte $03,$00,$06,$06,$0e,$0e,$03,$03
.byte $01,$06,$06,$0e,$0e,$03,$03,$01
.byte $01,$06,$0e,$0e,$03,$03,$01,$01
.byte $0f,$0e,$0e,$03,$03,$01,$01,$0f
.byte $0f,$0e,$03,$03,$01,$01,$0f,$0f
.byte $0c,$03,$03,$01,$01,$0f,$0f,$0c
.byte $0c,$03,$01,$01,$0f,$0f,$0c,$0c
.byte $0b,$01,$01,$0f,$0f,$0c,$0c,$0b
.byte $0b,$01,$0f,$0f,$0c,$0c,$0b,$0b
.byte $00,$0f,$0f,$0c,$0c,$0b,$0b,$00
.byte $0b,$0f,$0c,$0c,$0b,$0b,$00,$0b
.byte $0b,$0c,$0c,$0b,$0b,$00,$0b,$0b
.byte $05,$0c,$0b,$0b,$00,$0b,$0b,$05
.byte $05,$0b,$0b,$00,$0b,$0b,$05,$05
.byte $03,$0b,$00,$0b,$0b,$05,$05,$03
.byte $03,$00,$0b,$0b,$05,$05,$03,$03
.byte $0d,$00,$00,$00,$00,$00,$00,$0d //25
.byte $01,$00
```

```
.pc = timing "timing"
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
```

```
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07
.byte $00,07,$07,$07,$07,$07,$07,$07 //25
.byte $00,07
```

.pc = pattern “pattern”

```
.byte $06,$06,$0e,$0e,$03,$03,$0d,$0d
.byte $07,$07,$0f,$0f,$0a,$0a,$02,$02
.byte $00,$0b,$0b,$0c,$0c,$0f,$0f,$01
.byte $01,$0f,$0f,$0c,$0c,$0b,$0b,$00
.byte $06,$06,$0e,$0e,$03,$03,$01,$01
.byte $03,$03,$0e,$0e,$06,$06,$00,$06
.byte $06,$0e,$0e,$03,$03,$0d,$0d,$07
.byte $07,$0f,$0f,$0a,$0a,$02,$02,$00
.byte $06,$06,$0e,$0e,$03,$03,$01,$01
.byte $0f,$0f,$0c,$0c,$0b,$0b,$00,$0b
.byte $0b,$05,$05,$03,$03,$0d,$0d,$07
.byte $07,$03,$03,$05,$05,$0b,$0b,$00
.byte $06,$06,$0e,$0e,$03,$03,$0d,$0d
.byte $07,$07,$0f,$0f,$0a,$0a,$02,$02
.byte $00,$0b,$0b,$0c,$0c,$0f,$0f,$01
.byte $01,$0f,$0f,$0c,$0c,$0b,$0b,$00
.byte $06,$06,$0e,$0e,$03,$03,$01,$01
.byte $03,$03,$0e,$0e,$06,$06,$00,$06
.byte $06,$0e,$0e,$03,$03,$0d,$0d,$07
.byte $07,$0f,$0f,$0a,$0a,$02,$02,$00
.byte $06,$06,$0e,$0e,$03,$03,$0d,$0d
.byte $07,$07,$0f,$0f,$0a,$0a,$02,$02
.byte $00,$0b,$0b,$0c,$0c,$0f,$0f,$01
.byte $01,$0f,$0f,$0c,$0c,$0b,$0b,$00
.byte $06,$06,$0e,$0e,$03,$03,$0d,$0d
.byte $07,$07,$0f,$0f,$0a,$0a,$02,$02
.byte $00,$0b,$0b,$0c,$0c,$0f,$0f,$01
.byte $01,$0f,$0f,$0c,$0c,$0b,$0b,$00
.byte $06,$06,$0e,$0e,$03,$03,$01,$01
.byte $0f,$0f,$0c,$0c,$0b,$0b,$00,$0b
.byte $0b,$05,$05,$03,$03,$0d,$0d,$07
.byte $07,$03,$03,$05,$05,$0b,$0b,$00
```