Demo Coding

This section contain material related to demo coding. General VIC programming is not available on this page. Instead it is devoted to coding specific demo effects and tutorials on how to make demos specifically (rather than coding in general). General drawing, non-specific to demo coding, is also on the general VIC page instead.

To give an example: Opening the border is commonly made in demo parts, but simply opening the border does not constitute a demo effect in itself (at least not during the last 20 years).

General Information and Tutorials

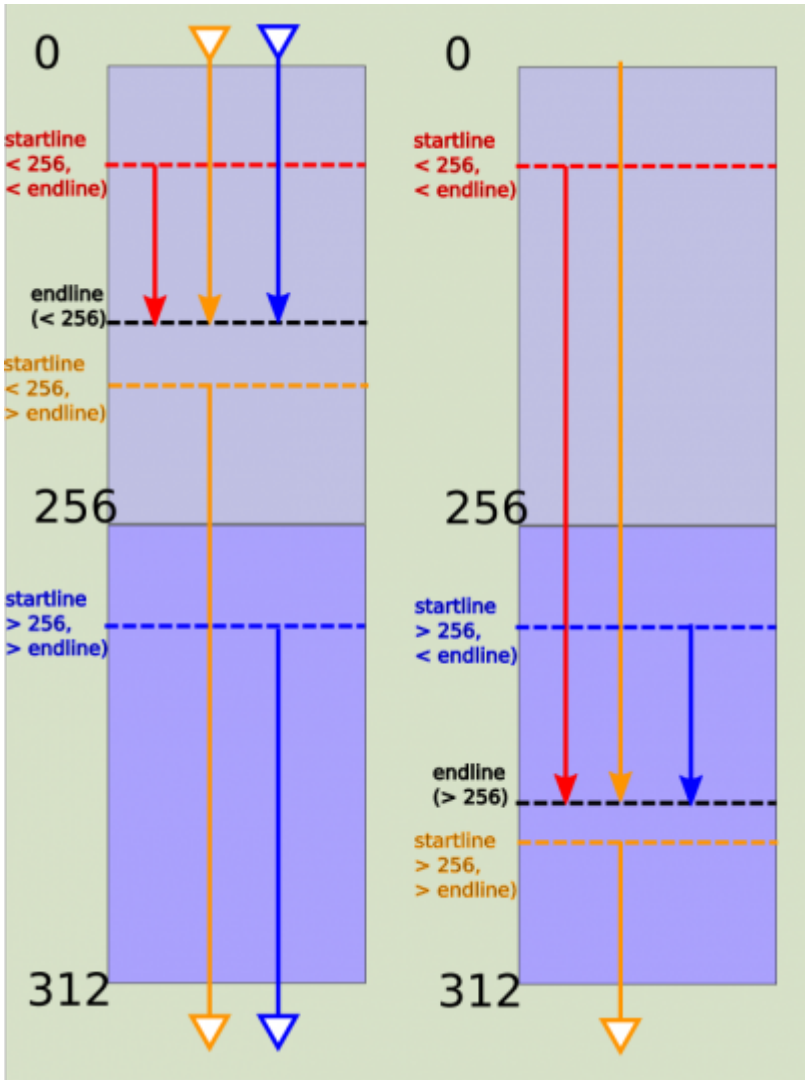
- An Introduction to Programming C-64 Demos - Puterman/FLT. Extensive and easy to read tutorial aimed at beginners.
- Demo world records and world firsts - Who broke what record? Who invented effect X first? When? In what demo?
- Speeding up and optimising demo routines - by conrad

Effects

Many demo effects depend on precise timing. Therefore it might be useful to have a look on the Interrupts and Timing page.

Horizontal Rastersplits

Very basic rastersplitting can be done by waiting for a certain rasterline an then switch some VIC register. The display has 312 rasterlines at max but the VIC rasterline register ($d012) has only a range of 0 to 255. Therefore bit 7 of $d011 indicates if the rasterline is <256 or >255. Now the questions arises how to wait for a certain rasterline in the whole range without the preassumption of beeing in rasterline x when the wait routine is called. Here is a diagram which examples the situation:



Dependent of which rasterline you're currently in, you can simply compare $d012 with the desired rasterline or you must first wait for bit 7 of $d011 to attain the right value and then wait again for the desired line. Even more, retrace has to be taken into account, too.

In total you have to consider 6 different cases:

being in rasterline 0-255 and

- waiting for a rasterline < $d012 → wait for bit 7 of $d011 to switch to 1 and back to 0 and then wait for lowbyte
- waiting for rasterline > $d012 but < 256 → simply wait on lowbyte to match
- waiting for a rasterline > $d012 but > 255 → wait for set bit 7 of $d011 and then wait for lowbyte to match

being in rasterline 256+ and

- waiting for a rasterline < $d012 and < 256 → wait for unset bit 7 of $d011 and then wait for lowb
- waiting for rasterline < $d012 but > 256 → wait for unset bit 7 of $d011, then set bit7 of $d011 and then on lowbyte
- waiting for a rasterline > $d012 → simply wait for lowbyte to match

```
waitrasterline:
```

```
    cpx #0
    beq wait0To255
    ;from here on we wait for a rasterline > 255
    bit $d011
    bpl *-3
    ;inRasterLineGT255
    cmp $d012
    bcs waitMatchingD012
    bit $d011
    bmi *-3
    bit $d011
    bpl *-3
    bmi waitMatchingD012
wait0To255:
    bit $d011
    bmi *-3
    ;inRasterlineLT256
    cmp $d012
    bcs waitMatchingD012
    bit $d011
    bpl *-3
    bit $d011
    bmi *-3
waitMatchingD012:
    cmp $d012
    bne waitMatchingD012
    rts
```

Rasterbars

The most classic demo effect, apart from scrolling text.

- Rasters - what they are and how to use them (C=Hacking #3) - by Bruce Vrieling - (Note that there is a bug in the last example program that turns border black and white: the high and low bytes of “intcode” are reversed in the lda instructions. Also note that on PAL machines the refresh rate is not 60hz, but 50 hz.)
- Rasterbars source - by Knoeki
- Rasterbars 2 source - by Bitbreaker
- Rasterbars 3 source - by Graham
- Rasterbar Flasher source - Flash Screen effect by Wozza/CygnusOz

Scrolling text

Those scrolling texts that we all hate to love that we love to hate. Also see the sprite section for a sprite scroller.

- Scroll text in common - by Monte Carlos

- 1 char sized scrollers:
  - Scroll text - by Vai/Slash Design
  - Scroll text - By Richard Bayliss/TND
  - Scroll text - variable speed and direction, by Groepaz/Hitmen
- Zoom chars 8x:
  - Char zoom - by Conrad.
  - Char zoom - by Raf/Vulture Design
- Sprite scroller:
  - Scrolltext using Sprites - by Testicle
- Perspective Scroller
  - Discofloor scroller as being used in Ächzzeit - by Bitbreaker/Oxyron^Arsenic^Nuance

Swinging and tech-tech

- TechTech (or “wave”) - by Pasi 'Albert' Ojala (from “Demo corner” in C= Hacking 7).
- TechTech (using FLI routine) - by Compyx/Focus
- Logo swing - By Richard Bayliss

3D dot scroll

- 3D Dot Scroll - by wegi /Black Sun/Samar/Fatum

DYCP

- DYCP - Pasi 'Albert' Ojala (from “Demo Corner” in C= Hacking 6).

DYSP

- DYSP using sprite stretching - by Compyx/Focus
- DYSP using a cycle table - by Compyx/Focus

Plasma

- ColorCyclePlasma - By Cruzer
- AFLI-Plasma - by Testicle
- Proportional-Charset-Noter with Plasma-Effect - by Testicle
- Copper Style FLI Plasma - By Cruzer

FPP (Flexible Pixel Position, aka Stretcher)

- FLI-FPP-Scroller - by Testicle

Graphics Distortion

- FLI Floffy - By Cruzer
- 2nd Pixel FLI Distorter - By Cruzer

Fractals

- Julia Fractal Morpher - By dW

Vectors

- Drivecalc vectors - 3D realtime filled vectors with 3D calculations done in the drive, by wegi /Black Sun/Samar/Fatum
- Filling the vectors - by Bitbreaker
- Spritevectors - by Bitbreaker

Blending and Fading

- Blend Charsets by Chico /CIVITAS

Starfields

- 8 Sprite starfield - by Richard Bayliss
- ROL Starfield using $d018 - by Richard Bayliss

2nd Line FLI

- Twisters, x-rotators and waving carpets - Bitbreaker/Oxyron^Arsenic^Nuance

Fire Effects

- 4x4 charset fire with lots of colors - Bitbreaker/Oxyron^Arsenic^Nuance

Misc

- Colour flashing (notewriter style) - (Extended colour mode) by Richard Bayliss
- just Animation - explains the concept of how the animations in reanim8ed work. originally

published in Driven #31

Software screen modes for effects

- 16×16:
  - 16x16 char matrix by Monte Carlos
  - 16x16 Matrix Scroll by Chico /CIVITAS

Optimization

- Speedcode by Cruzer/CML
- Speeding up and optimising demo routines - by conrad
- Advanced optimizing - by Bitbreaker/Oxyron/Nuance