How to GoDot #6

By Arndt Dettke

Lettering Images (part 1)

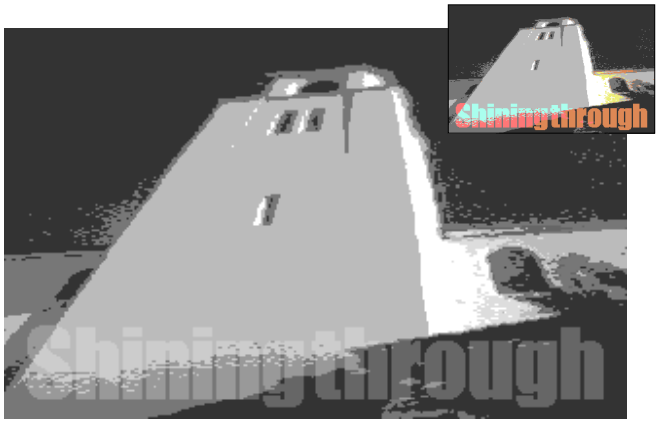
The October 2003 issue of the DIGEST opened with its main title created using GoDot. The letters were rimmed, and they had a floating shadow behind them. In this issue and in the following one I will cover some more tricks how to impressively letter your graphics with GoDot.

Prior condition (as long as we didn't talk how to "write" with GoDot) is that you have the titles you want to apply to your graphics available. They should be white letters on black background in GoDot's 4bit graphics format. You can create them with any Commodore paint application, preferably one which works in hires mode to gain smoother lettering, and load them to GoDot which will convert them to the wanted format at the same time.

The easiest way to apply lettering to your graphics is the overlaying method which we covered in H2G3 (September 2003). In short, this is the "***Compose***" mode of **ldr.4BitGoDot**, used with either option "***Foreground***" or "***Background***". Remember that black works as a transparency color in Compose mode, so overlaying white letters on black would mean using "*Foreground*" to apply these letters to your image.

We didn't yet use option "***Mix***" in Compose mode and its two parameters "***Percentage***" and "***Application Mode***". "*Mix*" is for merging two images together where you can control the visibility of the merged image by the percentage button (toggling between 25%, 50% and 75%). The middle button toggles between "***all***" and "***FGr***" which relates to color black being transparent ("*FGr*") or not ("*all*").

In any case, *Mix* results in grayscaled images of 16 different gray levels. Since a C64 is only able to display five true grays, GoDot uses colors to represent the missing 11 grays. This is the reason why GoDot's palette is differently ordered as compared to the standard C64 palette. We have ordered the colors to the brightness levels the VIC (the C64's video chip) provides. If you dial off the colors from your monitor while watching GoDot's palette requester you'll see what this means.



Pic #1: The letters have been mixed with 25%. Just viewable!

Anyway, white writing applied by *Mix* results in some colorful lettering which looks pretty nice in itself. If you like to have the real gray things (for printing purposes) you have to save the image to formats which are capable of many grays, such as GIF or EPS (see GIF in pic #1). However, you'll need an REU to use the according savers: **svr.GIF** and **svr.PostScript**. You can of course rim your semi transparent texts like described in the October issue of the DIGEST (H2G4).

The next example to add effects to letters is a slight 3D effect (a 1 pixel extrusion), looking very nicely (see pic #2). It works with heavily using **mod.Scroll** and **mod..Histogram** and the *Foreground* option of ldr.4BitGoDot.

Pic #2: Looking like thin plates lying on the image. This is a one-pixel extrusion with light from northwest.

First, you load the letter image and recolor white to any other color with mod..Histogram, option "***Swap***". Which color you choose doesn't matter since in the last step everything will be recolored to the desired look (however, this part of the letters will become the upper left rim, where the light "shines" on). Next, you displace the whole image by two pixels northwest (both two pixels left and up) with mod.Scroll. Then, you reload the image with option "*Compose – Foreground*". Again, you recolor white to any other color, producing the later lower right rim, the "shadow". Third step, you move the image one pixel back southeast, and reload the letters for the third time. Now you recolor everything: white to the wanted color of the letter plate (in pic #2 this is light blue), black to mid gray (this is because the shadow shall become black), the color of the first step to white (the light), and the color of the second step to black (the shadow).

Displaying this prepared letter image in multicolor mode will eventually look a bit "torn", but this is due to the 64's video limitations, a printout would show what you see here. Oh, don't forget! To overlay this lettering to an image (like Klimt's "Judith" in pic #2) you use the masking method of H2G5. Create a stencil mask from the background color with **mod.QuickMask** (thus defining it as transparent), and finally add the wanted image with **ldr.4Bit&Mask**.

Pic #3: This image imported from a digital camera and "bump mapped" with the famous name of the person in it.

The next, a bit more sophisticated way to letter images is the so-called "bump mapping". Yes, you read it right: with a bump map you produce bumps onto your images! ;-) Referring to letters this means you let the letters look as if they were coined into the image. It's sort of another 3D effect. Have an impression of it at pic #3.

To reproduce this effect you just need GoDot's Emboss modifier (**mod.Emboss**) and **ldr.4Bit&Map**. Load your title graphics and instantly apply mod.Emboss. This results in a most 3D looking title writing on a mid gray background. Now you install ldr.4Bit&Map, execute it and click "***Modify Data***". You'll get a new dialogue box in which you select "*Apply as:* ***Bump Map***". When prompted with the file selection box you just choose the file you want to apply the bump map to, and here we go.

Note that the letters in pic #3 look a bit darker than after the above directives. This is because I additionally darkened the portion of the image within the letters as described in H2G1 (July 2003) and then re-added it by masking the letters out (see H2G5, November 2003).

Enough for this time, folks!

Ldr.4BitGoDot – Default GoDot loader. Offers options to process images while being loaded ("***Compose***"). Option "***Background***" loads an image only to places where in the current image are black pixels (black is transparent in the current image). Option "***Foreground***" loads only those pixels from disk that are **not** black (black is transparent in the image being loaded). "***Mix***" merges images by the amount the percentage button says (***25%, 50%, 75%***). This amount controls the visibility of the image being loaded. "***FGr***"/"***all***" toggles the transparency of black during Mix (off/on). Mix results in **grayscale** images using colors as representations of lacking grays.

Ldr.4Bit&Map – Another loader to retrieve 4Bit images and to process them during load. In the "***Modify Data***" section you have three options: "***Addition***", "***AlphaCh(annel)***", and "***Bump Map***". Bump Map uses a gray scale image as a means to apply 3D-like distortions to the image in memory (see Pics #3 and #4). The other options will be covered in a future issue.

Mod.Emboss – Modifier which belongs to the edge detecting graphics filters. It finds every northwest-southeast diagonal edge in the image and very much intensifies it. Everything else in the image gets discarded and replaced by mid gray (color 7 in GoDot). A nice effect that turns images into reliefs. Use mod.Convolve to define and apply edge detectors for other directions.

Command history

For Pic #1:

(Load: 4BitGoDot)
Load Replace “menace.4bt”
Load Compose
Mix: FGr 25%
Mix “shiningtitle.4bt”
Display

Save this image as a grayscale GIF:

Save “GIF”
Save “shiningthru”
Select Palette: Gray16
(Save Area: Full)
Save Image

For Pic #2:

Load Replace “gkshape.4bt”
Inst: .Histogram
Execute
Swap wht, lgr
Exit
Inst: Scroll
Execute
Set Amount: 2
Direction: Left
Execute
Direction: Up
Execute
Leave
Load Compose Foreground “gkshape.4bt”
Inst: .Histogram
Swap wht, yel
Exit
Inst: Scroll
Execute
Set Amount: 1
Direction: Right
Execute
Direction: Down
Execute
Leave
Load Compose Foreground “gkshape.4bt”
Inst: .Histogram
Swap wht, lbl
Swap blk, gr2
Swap lgr, wht
Swap yel, blk
Exit
Inst: QuickMask
(Select:) gr2
Generate
Leave
Load: 4Bit&Mask
Get 4Bit “klimt.4bt”
Leave
Display

For Pic #3:

Load Replace “kdale.4bt”
Inst: Emboss
Execute
(Display)
Load: 4Bit&Map
Load Modify Data
Bump Map “doubledale.4bt”
Display

Have ever fun using GoDot!

Pic #4: Something like a stamp „bumped“ onto this image.